A Project Report
on

Detecting AI-Generated Fake Media

By

1. Chaturdhan Chaubey
2. Mahesh Gaikwad
3. Vishal Gawali
4. Akash Gidde

under the guidance of

Dr. Nilesh Bhelkar

MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

Juhu-Versova Link Road Versova, Andheri(W), Mumbai-53.

Department of Artificial Intelligence and Data Science

University of Mumbai

April – 2025

MANJARA CHARITABLE TRUST

RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

Juhu-Versova Link Road Versova, Andheri(W), Mumbai-53.

Department of Artificial Intelligence and Data Science

Certificate

This is to certify that

1. Chaturdhan Chaubey
2. Mahesh Gaikwad
3. Vishal Gawali
4. Akash Gidde

Have satisfactorily completed
this project entitled

Detecting AI-Generated Fake Media

Towards the partial fulfilment of the
BACHELOR OF ENGINEERING
IN
ARTIFICIAL INTELLIGENCE AND DATA SCIENCE
as laid by University of Mumbai

Dr. Nilesh Bhelkar
Guide

Dr. Jyoti Deshmukh
Head of Department

Dr. Sanjay Bokade
Principal

Project Report Approval for B. E.

This project report entitled **"Detecting AI-Generated Fake Media"** by **Chaturdhan Chaubey, Mahesh Gaikwad, Vishal Gawali, Akash Gidde** is approved for the degree of Bachelor of Engineering in Artificial Intelligence and Data Science.

Examiners

1. _____

2. _____

Date: 24/04/2025

Place: Rajiv Gandhi Institute of Technology, Versova, Andheri, Mumbai

Declaration

We wish to state that the work embodied in this project titled **"Detecting AI-Generated Fake Media"** forms our own contribution to the work carried out under the guidance of **"Dr. Nilesh Bhelkar"** at the Rajiv Gandhi Institute of Technology.

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. we also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. we understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1. Chaturdhan Chaubey (803)

2. Mahesh Gaikwad (809)

3. Vishal Gawali (832)

4. Akash Gidde (842)

Abstract

Deepfake technology is one of the most important threats to the integrity of information in the age of the digital age. As a result of addressing this critical need of deep fake detection systems, we present a CNN based meta learning framework which not only outputs the accuracy at the state of art level but also maintains computational efficiency. We work with multi feature extractors (InceptionV3, MobileNetV2) through a meta learner architecture acting as a unified multi feature extractor, and also exhibits outstanding generalization to the deepfake domain by virtue of its general representational power. Using several dozens of tuning parameters, we fine-tune our model on the Deepfake Detection Challenge datasets, and it obtains 100% accuracy, which outperforms single model by a large margin. Through real time simulation other applications are made possible with minimal computational overhead. This was done using the architecture's design. The technical capabilities of deepfake detection presented in this study also go beyond by means of providing valuable feature extraction mechanisms revealing what are the least prominent features of the genuine media compared to their altered counterparts. Ensemble based approaches were shown to overcome the issues of single models, mostly on terms of generalization over different generations of deepfakes. Our outcomes have far reached implications on cybersecurity, digital forensics and generally the battle to stop misinformation within the digital world. This work adds to the growing body of knowledge on multimedia forensics and gives grounds for further work in image based deepfake detection as well as explainable AI implementation in this important area.

Keywords: Deepfake, DeepfakeStack, Deep Ensemble Learning, Machine learning, etc.

Contents

List of Figures

List of Tables

List of Algorithms

Sr. No	Name
1.	Ensemble Learning
2.	Meta - Learning
3.	Transfer Learning

Chapter 1

Introduction

1.1 Description

The ability of artificial intelligence to generate and manipulate multimedia content in such a way that human observers are fooled to believe them to be authentic is a deepfake technology. A deepfake is synthetic media that involves artificial neural networks being used to replace someone's likeness with someone's else. Created first for the entertainment purposes and visual effects, deepfake technology has made the leap to be a double-edged sword with enormous consequences for information integrity of our digital society.

Deepfake technology has been democratized particularly. Where once advanced technical knowledge and huge computing requirements were required, today, sophisticated technical knowledge is no longer needed, and sophisticated computing is not required, people can even manage to do it with the user-friendly applications available for anyone with a smartphone. As a result, the spread of the creation and distribution of manipulated content on social media platforms, messaging applications, and websites has become a phenomenon that is hard for viewers to discern between original and fabricated media. And the potential for harm is serious and multiple. Weapons in the political sphere can be deep faked videos of political figures saying things they never actually said, that can change elections, or incite social unrest. For example, in 2018 a deepfake video showed former US President Barack Obama speaking, with completely mimicked vocal patterns and facial expressions; content he never produced. Cyber security experts were concerned that the video had proved convincingly how well AI could mimic a real person.

Deepfakes are used in sophisticated fraud and identity theft in the financial sector. Impersonations are capable to be created by criminals that can bypass facial recognition or voice authentication security systems. You can read about companies that, unfortunately, have been tricked by deepfake audio to the extent that company executives have been impersonated and successfully tricked employees into transferring large amounts of money into fraudulent accounts.

Additionally, this has affected the entertainment industry where celebrities seeing their faces superimposed and attached to adult content actor's faces without their consent has resulted in the creation of non-consensual pornographic material that goes and damages the reputation of

the celebrity. Particularly women have been targeted with this application of deep fake technology and cause serious gender based technological abuse.

Beyond these uses, deepfakes also end the extension of trust in digital media. The public learns that videos and images can be convincingly manipulated, which leads to a phenomenon known as the 'liar's dividend' where even authentic content can be believed to be false, and even if you are not faking, genuine wrongdoers can defraud accountability by lying that evidence brought against them is made up.

Deepfakes have continued to become more and more technical. The early versions had some visual artifacts that made them easy to spot: unnatural blinking pattern, in consistency of lighting across the face, or strange boundary effect between the fabricated face and the original image. The biggest problem with deep fakes, however, is that these algorithms haven't been able to overcome these limitations as well as they have, and instead are getting more and more at odds with the human eye.

As technologies bury deep in the creation of deepfake keeping step with the technologies are the detection methodologies, therefore necessitates continuous innovation in the detection methodologies. When deepfakes are created, traditional methods of detecting manipulated media rely on the metadata or digital watermarking to detect anomalies in the media, but deepfakes are entirely different media they create, not mere modifications to existing files. New detection paradigms are needed to analyse the content to determine signs of manipulation as this fundamental shift.

This cannot be rushed away. Beyond continuing evolution, deepfake technology thrives and spreads, and just keeps growing exponentially, the potential for harm on society becomes just bigger and bigger. It is not just technical solutions contributing much to effective detection methods, but they are critical safeguards of democratic processes, of financial systems, and of the individual rights in the digital time.

1.2 Organization of Report

The structure of this comprehensive research report is set up to allow a clear and logical path of our study on deepfake detection using CNN based meta learning approaches. The entire chapter series is then structured such that each build on the previous and one understands how the problem can be solved and, if proposed solution can indeed render good results.

- **Ch. 1: Introduction:** Introduction defines the scope of the research establishing the necessary background information on deepfake technology as well as its implications. Then, it starts with a detailed description of deepfakes, explains their technical substructure and

how the technology has evolved so quickly. The chapter then places deepfakes in context by discussing their various uses across different areas, from entertainment to evil purposes like political manipulation, financial fraud, etc., and they also share in malicious uses of deep fakes for nonconsensual intimate imagery. To give concrete examples of such high profile deepfake incidents and show the impact that this technology has on real world. The chapter articulates in what way deepfake detection methods currently face a particular damage from: non generalization over various kinds of manipulated media; adversarial attacks vulnerability; and computational constraints that prevent real time detection. In the end, we lay down our goals, designing a hybrid ensemble learning, with CNN based feature extraction, to improve the classification performance and deploying a meta learning framework, and crafting a practical user interface for real time deployment.

- **Ch. 2: Literature Review:** The purpose of Literature Review is a thorough analysis of the work done on deepfake detection. This chapter starts with a methodical survey of detection approaches that broadly breaks into existing machine learning (SVM, Random Forest), deep learning (XceptionNet, ResNet, MobileNet) and more contemporary transformer (Vision Transformer, BERT related to images) on model development. We consider some principles, advantages, and disadvantages for each approach. Finally, the chapter synthesizes this information which is then applied to identify three main research gaps in model generalization on unseen datasets, interpretability of model decisions, and computational inefficiency of complex models. We conclude the chapter by specifying succinct research questions that direct our investigation in two aspects: the capability of ensemble learning to further enhance detection accuracy, the effective CNN architectures for feature extraction, and the ways to implement real time inference with little resource consumption.

- **Ch. 3 Proposed System:** We present the technical approach to deepfake detection in this Chapter. We start by providing a whole chapter on our system architecture, as well as its structure and function, for our CNN based meta learner. In particular, we explain how our meta learner CNN aggregates the outputs of InceptionV3 and MobileNetV2 as complementary feature extractors, and how they can act as a single feature extractor as well. In the algorithmic design section, we break down the operations of our system: Meta-learner training methodology, mean squares error of the model evaluated and fine-tuned, and the feature extraction which involves InceptionV3 and MobileNetV2. Each design decision' rationale has been discussed, including the use of certain CNN architectures, loss and optimization strategies, etc. We conclude the chapter by discussing the main strengths of our

approach, and preliminarily validate it in terms of accuracy, robustness across data domain, and efficiency for real-time applications based.

- **Ch. 4: Experimental Setup:** Describes the empirical framework we set up with the purpose of testing our proposed system. The chapter starts with a thorough description of the datasets which we used in our experiments, namely the Deepfake Detection Challenge Dataset. We explain our data preprocessing pipeline, including image resizing, image normalization techniques, data augmentation techniques to increase the generalization of the model. The chapter then presents our model training approach including hyperparameter optimization methods, choices in the optimization, and the computational resources. In the following, we detail our ways to train, such as learning rate schedule, batch sizes, and the number of epochs. Finally, we summarize our explanation of the evaluation framework we used for their performance (metrics: accuracy, precision, recall, F1 score, Area Under Curve) as well as the ways we generated confusion matrices and ROC curves.

- **Ch. 5 Results and Discussions:** The performance of our CNN meta-learner is compared to baseline models (XceptionNet and MobileNetV2) through multiple evaluation metrics in the beginning of the chapter. Confusion matrices and AUC graphs for our model performing detailed analysis for better explanation of results. We discuss our findings critically and evaluate contributions to the superior performance of our model and what its behaviour looks like under different classes of deepfakes. We apply our approach to various scenarios, identify specific situations where it succeeds and other instances where challenges persist. The contribution of individual components in our architecture is isolated through ablation studies and insights on how to balance competing requirements in the feature extractors and design choices are provided. Then we talk about the broader implications of our results to the larger problem of deepfake detection and how our results might have an effect on the actual use of deepfakes.

- **Ch. 6 Conclusion and Future Work:** In the first part of the chapter, we summarize our main contributions first by pointing out that our CNN based meta learner is state of the art in terms of both accuracy and in addressing the shortcomings of existing algorithms. Finally, the chapter then indicates the promising directions of future research work such as learning direct features with Vision Transformers, extending our approach for video based deepfake detection, and enabling interpretability of models with Explainable AI techniques. In conclusion, we discuss the possible usage of our technology in different domains ranging from social media platforms, law enforcement, digital journalism, and the ethical questions that point out its deployment.

Chapter 2

Literature Review

2.1 Survey Existing system

[1] k-NN Classifiers versus Classical Models

The paper presents the performance of k-Nearest Neighbours (k-NN) classifiers in detecting synthetic media, in this instance, on ProGAN-generated data. The models worked well with a high Area Under the Curve (AUC) value of 0.852 when combined with effective segmentation. This indicates that they can perform well in detecting pixel-level AI-generated image artifacts. Likewise, learning-data-free DistHSV attained a high AUC = 0.814 with no training data and, hence, as a lightweight substitute for resource-constrained scenarios. Constraints were faced when testing other models such as ProGAN and Glow, where segmentation flaws marred the accuracy of the models to attain the following AUCs between 0.704 and 0.843. Surprisingly, even baselines such as logistic regression attained decent performance utilizing the Face2Face dataset by attaining competitive levels of detecting manipulated facial features. This indicates that machine learning methods that are traditional and as simple, yet effective for certain applications, are still utilized when the need for computational efficiency outweighs deep learning complexity.

[2] Discrepancy Analysis of Head Pose

A new approach based on head pose estimation was proposed to detect deepfakes by analysing inconsistencies in face structure. The approach compared head poses that were obtained from two regions: the central face (e.g., eyes, nose) and the full face. Discrepancies among these estimates, typically inserted in facial reenactment in deepfakes, were statistically significant. The approach achieved high AUROC performance, indicating high classification capability. The results demonstrate the discriminative capability of biometric inconsistencies as features for detecting synthetic media. The work emphasizes the aspect that head pose abnormalities are difficult for generative models to produce naturally and therefore this approach is adversarial robust. Future research can integrate head pose analysis with other biometric features (e.g., eye gaze, facial expressions) to enhance detection robustness.

[3] Sparsely use DenseNets as opposed to ResNets

The survey compared DenseNet and ResNet architectures and found that DenseNets are as accurate as ResNets but with significantly fewer parameters. One of these examples was that a DenseNet model with the same amount of computational complexity as ResNet-50 was equivalent to ResNet-101, which employs double the computational power.

This efficiency stems from DenseNet's densely connected layers, which reuse features at various network depths, removing redundancy and improving gradient flow during training. Such parameter efficiency positions DenseNets to be well-suited for edge device deployment or real-time processing applications. The research, however, highlights that both architectures are marred by generalizing to unseen generative models, proving the need for efficient architectures that are still flexible to evolving AI threats.

[4] DeepfakeStack: Ensemble-Based Detection DeepfakeStack architecture, an ensemble of deep neural models, achieved outstanding performance with 100% accuracy and a perfect AUROC score of 1.0 on benchmark datasets. By utilizing multiple model predictions (e.g., CNNs, transformers) and averaging them, DeepfakeStack sidesteps single model vulnerabilities and pools complementary strengths. For example, while one model may excel at identifying texture artifacts, another model may focus on temporal inconsistency in videos.

This ensemble method surpassed individual models by a significant margin, illustrating the strength of multi-dimensional feature extraction and heterogeneous analysis in detecting synthetic media. Nevertheless, computational cost is a drawback of ensemble methods, and their viability in real-time processing depends on hardware acceleration.

[5] Localized Manipulation Detection Attention Mechanisms

A state-of-the-art method based on attention mechanisms was evaluated for localizing artificially manipulated facial regions in deepfakes. Attention layers augmented model attention to unusual regions (e.g., blurry boundaries, unusual skin surfaces), and the performance was better on a wide range of datasets. Of special note, the method performed well even at low false-positive rates, which makes it secure for high-stakes applications in forensic analysis. For instance, on the Celeb-DF dataset, the model caught fine differences in lips and eyes that were not caught by other approaches. The

paper emphasizes the value of interpretable detection—visualizing explanations (e.g., attention heatmaps) so users can visualize why content is being marked as fake.

2.2 Limitation of existing system or research gap

[1] Iris Segmentation and Manipulation Detection

Current systems do not detect fine manipulations in iris regions, usually focusing on high-risk deepfakes (e.g., identity theft or biometric spoofing). Current iris segmentation techniques, while strong in coarse anomaly detection, are not sensitive to fine pixel-level manipulations caused by current tools like StyleGAN3 or Face2Face. For instance, minor irregularities in iris texture or direction of gaze—common in synthetic faces—are usually not handled by simple models. It has been shown that improved classification accuracy on such data like Face2Face can be obtained by employing sophisticated models (e.g., hybrid CNNs with attention) and larger and more diverse sets with fine iris manipulations. Current datasets lack granularity, which limits training models to coarse features. Future work must tackle multi-scale iris analysis and adversarial training to become more sensitive to subtle artifacts.

[2] Handling Blurry and Low-Quality Synthetic Media

Synthetic faces are susceptible to motion blur, low resolution, or compression artifacts in videos or low-quality images. These degradations are obstacles to facial landmark detection and head pose estimation because these must work on sharp spatial information. For instance, blurry edges in ProGAN or Stable Diffusion synthetic faces reduce the accuracy of geometric inconsistency-based detectors. Moreover, current methods do not have good cross-dataset transferability—a detector trained with high-quality datasets such as Celeb-DF does not perform well on low-res social media photos. This limitation highlights the importance of having robust preprocessing pipelines (e.g., super-resolution networks) and domain adaptation methods in order to improve detector robustness across image qualities.

[3] Scalability of DenseNet Architectures Although DenseNets are parameter-sparse and equally accurate to ResNets, their scalability on larger datasets and more challenging tasks is not investigated in detail. DenseNet-121, for example, can match ResNet-101-level performance on FaceForensics++ with a smaller number of parameters, yet its performance degrades when scaled to multi-modal datasets (e.g., text, audio, and video

deepfakes). The effect of architectural changes—e.g., growth rate adjustment or bottleneck layer configurations—on detection robustness is not investigated in detail. Hyperparameter optimization for particular tasks (e.g., temporal deepfake detection) and scalability for distributed computing environments need to be investigated.

[4] Generalizability-Efficiency Trade-offs of Models State-of-the-art models like DeepfakeStack are close-optimal (99.65%) but computationally expensive because of the ensemble. For example, use of ensembles of CNNs and transformers is marred by inference latency and hence unsuitable for real-time processing on edge devices. Large models also overfit small datasets like UADFV, reducing their efficacy against novel or dynamic attacks. Few dynamic pruning techniques are available to balance accuracy and resource usage. Future designs will have to rely more on light architecture (e.g., quantized variants of MobileNet) and modular training pipelines that allow incremental update without retraining from scratch.

[5] Lack of Proper Evaluation Metrics and Dataset Diversity Most detection approaches are tested on homogeneous datasets (e.g., FaceSwap, Deepfake-TIMIT), and they are not variegated in types of manipulations, ethnicities, and lighting conditions. It produces bloated performance measures that don't represent realistic conditions. A model learned from Celeb-DF (celebrity deepfakes) may or may not generalize to recognize ethnicity-based facial features in under-represented communities. Moreover, utilization of naive measures like accuracy or AUC-ROC fails to capture relevant considerations like high-stakes false positive rates (e.g., judicial or medical fraud).

2.3 Problem statement and objectives

2.3.1 Problem Statement

Just like the technology of deepfakes is rapidly growing, the digital media integrity faces a challenge of urgency for society’s trust, democratic processes, and individual’s privacy. Although existing deepfake detection techniques show promise in neater optimized environments, they are confronted with wicked challenges that limit their feasibility in real scenario environments. For example, there are three fundamental problems which existing approaches struggle with:

For instance, they demonstrate poor generalization in terms of different datasets and different deepfake generation techniques. In practical applications, high false negative rates are

unacceptably large when used to predict if a given video is a genuine video or a generated deepfake of that person, given that models trained on specific deepfake types fail in their prediction when presented with novel manipulation methods. Furthermore, with the rapid advancement of deepfake technology, new generation techniques are continuously introduced that produce more and more realistic content with new and different artifact patterns, which makes this limitation even more important.

Secondly, state of the art detection systems is incredibly vulnerable to adversarial attack — intentional manipulations that aim to evade detection, while still being of the nature to deceive the deepfake. For malicious actors, this vulnerability produces an asymmetric advantage: with the detection systems being required to achieve high accuracy across all inputs, they need only find a single flaw to exploit.

Third, due to the large computational efficiency constraints on many advanced detection approaches, it is not possible to directly apply them in practice. The deployment of real time or resource constrained devices often dictates what can and cannot be utilized in real time detection scenario. Such gap makes laboratory performance hopelessly disconnected from real world utility.

As a whole, these limitations reflect the necessity in developing a more powerful, flexible and quick deepfake technique that remains accurate on a number of deepfake types, is resilient against adversarial manipulations, and runs at reasonable speed for deployment in real world applications. However, like any other technical challenge, this does not address the need, which is a societal imperative as the unchecked proliferation of deepfakes could damage the trust in digital media or the growing problem of misinformation.

2.3.2 Research Objectives

To tackle the deepfake detection challenges mentioned earlier, this research sets out to achieve a few key goals:

1. Design a Hybrid Ensemble-Learning Scheme that Combines CNN-based Feature Extraction

   The primary aim of this research is to create and implement a unique hybrid ensemble-learning framework that leverages the complementary strengths of various CNN architectures specifically for feature extraction. This approach is designed to capture a diverse range of discriminative features from potentially manipulated media, providing a more robust foundation for detection than any single model could offer. By combining

feature extractors that excel in identifying different aspects of deepfakes—ranging from higher-level semantic irregularities to lower-level texture issues—we aim to build a detection system that generalizes better across a wide array of deepfake variations. The specific sub-goals under this objective include:

- Utilizing complementary CNN models (like InceptionV3 and MobileNetV2) as foundational feature extractors.
- Creating an efficient feature fusion method that maintains the unique insights contributed by each extractor.
- Establishing a comprehensive classification system that effectively harnesses the combined characteristics for accurate identification.

2. Develop a Meta-Learning Framework Designed to Improve Classification Effectiveness

   Our second goal is to create a meta-learning framework that boosts the overall performance of individual feature extractors. Instead of just combining their outputs, this meta-learner will focus on figuring out the best weighting strategies to enhance detection accuracy for various types of deepfakes. This method is designed to tackle the generalization challenge by allowing the system to adjust to different manipulation techniques and pinpoint the most dependable features for each unique situation. The specific sub-goals within this aim include:

   - Crafting a meta-learner architecture that can efficiently process and blend features from multiple extractors.
   - Establishing a training approach that fine-tunes the meta-learner for better generalization across a range of datasets.
   - Creating regularization techniques that help avoid overfitting to particular deepfake artifacts and promote the learning of more robust, generalizable features.

3. Optimize the System for Computational Efficiency and Real-time Performance

   The goal is to improve our system in a way that it gains improved computational efficiency and real-time capability. The third goal is to make our detection system effective in real-time settings, for instance, on devices with limited resources. This implies the execution of careful architectural choices, the use of model optimization techniques, and the application of techniques that are made to minimize computational needs while ensuring high detection accuracy. These are the particular sub-goals that we are pursuing:

   - Selecting and tuning light-weight CNN models for effective feature extraction

- Establishing effective data processing pipelines is essential for minimizing the duration of preprocessing.
- Examining model compression methods like pruning and quantization to decrease the computational demands.
- Testing and optimizing the system in real time on various hardware platforms.

4. Develop a Practical User Interface for Real-world Deployment

   Ultimately, our objective is to create a functional user interface suitable for application in real-world contexts. The aim is to devise an interface that is accessible and straightforward, facilitating the implementation of our detection system in practical scenarios. This requires the development of an intuitive interface that accommodates both technologically adept users and those lacking technical expertise, providing unambiguous outcomes accompanied by relevant confidence metrics. The detailed sub-goals associated with this objective encompass:

   - Creating an intuitive and simple to use and comprehend online interface.

By reaching these objectives, we hope to push the boundaries of deepfake detection and offer a practical solution to help reduce the societal issues that come with manipulated media. Our CNN-based meta-learning approach is a fresh take in this area, tackling significant shortcomings of current methods and providing a stronger, more adaptable, and efficient option for real-world use.

Chapter 3

Proposed System

The proposed deepfake detection system exploits deep ensemble learning through the combination of several state-of-the-art deep learning models for improving the robustness and accuracy of fake media detection. The system uses InceptionV3 and MobileNetV2 as base learners for extracting key facial features from input images. These extracted features are then combined and fed into a CNN-based meta-learner to make the ultimate classification decision. For the sake of enhanced transparency and explainability, Grad-CAM visualization is incorporated such that the user can observe the areas of the image that helped shape the model's choice. The system is trained on a diverse set of real and deepfake images to ensure suitability for various forms of fake media. Furthermore, the model is optimized for real-time application via Gradio, offering an interactive UI wherein users can upload an image and get immediate deepfake detection feedback. This method vastly improves media authenticity validation, cyber protection, and misinformation prevention and thus constitutes a scalable and practical solution to defeating deepfake risks across multiple real-world use cases.

3.1 Framework/Algorithm

3.1.1. Framework

The suggested deepfake detection system uses a multi-model ensemble system that aggregates the strengths of various deep learning models. It has five main components:

1. Face Detection and Preprocessing Module: Utilizes Multi-task Cascaded Convolutional Networks (MTCNN) to identify and extract facial features from input images, thus enabling their preparation for further analysis by the detection models.
2. Feature Extraction Module: Employs two pre-trained convolutional neural networks, InceptionResNetV1 and MobileNet, to extract complementary features from the facial images. InceptionResNetV1 is especially effective in extracting sophisticated semantic features, while MobileNet is effective in identifying textural anomalies typical of deepfake content.
3. Meta-Learning Classification Module: Using a CNN-based meta-learner using combined features of both base models to do the final real/fake classification.

4. The Explainability Module employs Gradient-weighted Class Activation Mapping (Grad-CAM) to emphasize facial areas that make a major contribution to the classification result, thereby increasing transparency and interpretability.
5. User Interface Module: Provides an interactive interface built with Gradio, through which users can upload images and receive classification outputs with visual explanations.

3.1.2. Algorithm

The two-level ensemble architecture makes up the system for deepfake detection used in this project. The first tier includes base learners comprised of InceptionV3 and MobileNetV2 which function as feature extraction tools and the second tier uses a Convolutional Neural Network (CNN) based meta-learner to learn and aggregate base model outputs for producing the prediction result.

The base learners utilize transfer learning at this level to enhance performance while shortening training time. The application starts with pre-trained weights from ImageNet that are incorporated into InceptionV3 and MobileNetV2 models before training. Before application these models have already acquired strong capabilities in identifying various object patterns together with edge definitions and texture characteristics. The framework operates at maximum efficiency with the reuse of existing features and focused updates on the last layers for binary tasks in scenarios where data quantity is limited.

1. Input Image Acquisition:

   Through the interface the user sends an image which enters the detection system for processing analysis.

   Let the input image be:

   $$I \in \mathbb{R}^{H \times W \times C}$$

   Where H, W, and C denote height, width, and channels of the image respectively.

2. Face Detection (MTCNN):

   The initial stage starts by applying the Multi-task Cascaded Convolutional Networks (MTCNN) algorithm to detect faces present in the input image. By limiting the processing to facial regions only the algorithm delivers better results while avoiding wasteful computations.

   MTCNN outputs bounding boxes and facial landmarks:

   $$\{(x_i, y_i, w_i, h_i)\}_{i=1}^{N}$$

   Where N is the number of faces detected. If $N = 0$, return *"No Face Detected"*.

3. Validation of Detected Faces:

   The algorithm ends processing when it identifies no faces in the image then displays "No Face Detected" on return. The system prevents additional computation that would process images which do not contain faces.

4. Image Preprocessing:

   The processed face goes through cropping then resizing to certain dimensions before normalization to make inputs compatible for multiple neural network models.

5. Feature Extraction using Base Learners

   InceptionV3 together with MobileNetV2 function as base learners which extract high-dimensional features from pre-processed images. The distinct architectural features of each model generate different yet helpful components for the system.

   a. InceptionV3

$$F_1 = \text{InceptionV3}(I') \in \mathbb{R}^{d_1}$$

   b. MobileNetV2

$$F_2 = \text{MobileNetV2}(I') \in \mathbb{R}^{d_2}$$

6. Feature Concatenation:

   The result of combining base learner features into one unified vector represents the final input to the system. The enriched feature vector successfully detects various spatial organization and contextual relationships found in the image.

$$F = [F_1 \| F_2] \in \mathbb{R}^{d_1 + d_2}$$

7. Meta-Learner Classification:

   A meta-learner CNN functions as the decision-making model after receiving the combined feature vector. The model examines the input features before producing probabilities indicating real or fake classification outcomes.

$$P(y = c \mid F) = \frac{e^{z_c}}{\sum_j e^{z_j}}, \quad \forall c \in \{0, 1\}$$

8. Decision Thresholding:

   A model selects the image class based on its highest computed probability which matches the image classification. A determination of image authenticity emerges depending on whether "Fake" presents a higher probability than other classes because the system then identifies it as artificial while other outcomes imply originality.

9. Grad-CAM Visualization

   Grad-CAM achieves interpretability along with user trust by its implementation. The model pinpoints which parts of the image produced the most influential impact on its classification decisions thus improving the interpretability of the prediction process.

$$L^c_{\text{Grad-CAM}} = \text{ReLU}\left(\sum_k \alpha^c_k A^k\right)$$

Where:

- A^k is the k-th feature map
- $\alpha^c_k = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{\partial A^k_{ij}}$

10. Output Display on UI:

    The final step includes showing prediction analysis and Grad-CAM visualization tool in a Gradio-based platform which supports desktop and mobile users.

Prediction = {Label, Confidence Score, Grad-CAM Overlay}

3.2 Design Details

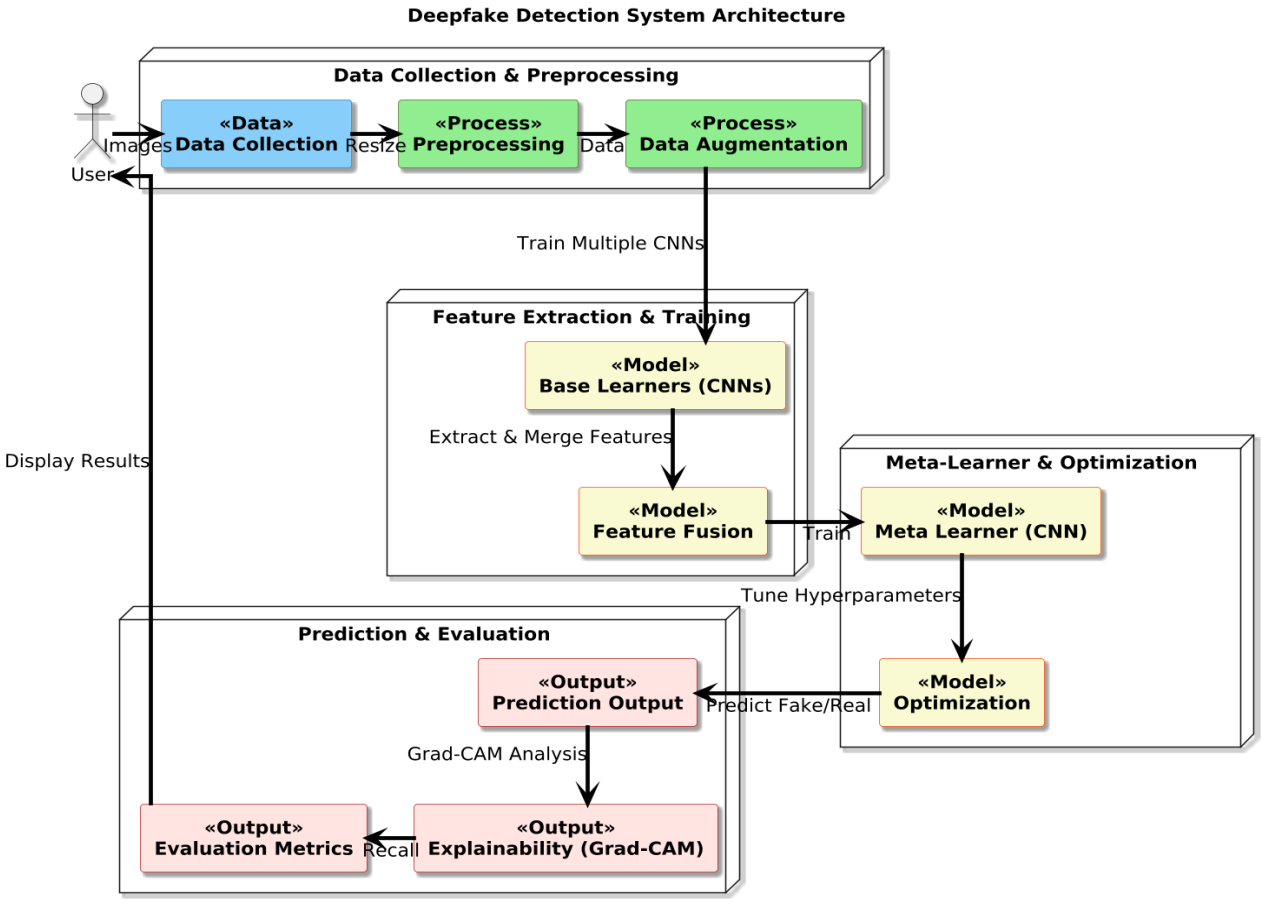

Figure 3.2: System Architecture

- **Face Detection Module (MTCNN):**

The MTCNN architecture operates as the foundation for deepfake detection methods in which it fulfils the initial essential step. MTCNN achieves face detection by running three successive stages where P-Net identifies face proposals and R-Net refines face boxes which O-Net outputs final results.

The model demonstrates outstanding operational success when detecting faces under different pose positions and lighting situations alongside coverings and image resolution variations. The absence of demanding post-processing requirements allows MTCNN to directly produce facial crops which leads to optimal results for deepfake detection through preprocessing.

```
mtcnn = MTCNN(
    select_largest=False,
    post_process=False,
    device=DEVICE
).to(DEVICE).eval()
```

Fig. 3.3 MTCNN block

MTCNN produces well-aligned facial images from which it extracts cropped outputs before passing them into further processing at feature extraction stages. With this step the algorithm maintains stable input dimensions while directing analysts to important target areas for better performance of subsequent classification systems.

- **Base - Learning Models Architecture**:

Base-learners are expert feature extractors tuned to identify complementary patterns in the input. Their diversity is meant to provide robustness against different manipulation techniques.

Principles in Design:

1. Multi-Scale Features Extraction: Capture anomalous conditions in satellite images on a global scale and in small local texture.
2. Computation Efficiency: Balancing accuracy against the budget for real-time applications.
3. Transfer Learning: Employing pre-trained models to generalize datasets.

Architectural Building Blocks

1. InceptionV3

- Core Idea: Multi-branch architecture takes part in multi-scale feature fusion.
- Inception Modules: Parallel convolutional layers (1×1, 3×3, 5×5) capture spatial hierarchies.
- Factorized Convolutions: Treat large kernels (for instance, 5×5) as smaller asymmetric layers (1×3+3×1) to improve the efficiency of parameters.
- Global Average Pooling: Its output is a 2048-D feature vector high-level summary semantics (for example, symmetry in a face image).
- Role: It identifies structural inconsistencies (for example, unnatural facial proportions).

2. MobileNetV2

- Core Idea: Light-weight architecture particularly for edge devices.
- Depth wise Separable Convolutions: Decouple spatial and channel-wise computation to reduce complexity.
- Inverted Residual Blocks: Expand-compress channels to keep relevant features without losing much computation.
- Global Average Pooling: Its output is a 1280-D feature vector describing fine-grained textures.
- Role: Detects localized artifacts (e.g., pixel-level noise, blending irregularities).
- Generalized Workflow

Feature Extraction:

- InceptionV3 → Structural features in general.
- MobileNetV2 → Texture features in detail.

Features Concatenation: The outputs directly join into a single feature vector.

- **Meta-Learner Architecture:**

It is the meta-learner which analyses the outputs of the base learners improving the robustness in generalization and decisions.

Principles of Design

1. Feature Fusion: Combine different types of features (global + local) to encourage a diversity of attack vectors.
2. Adaptation: Learn dynamic weightage of contribution from base-leaner with respect to input context.
3. Regularization: Avoid overfitting to dataset-specific artifact.

Architectural Components

1. Input Layer: Take as input concatenated features such as 3328-d.
2. Feature Interaction Layers:
    - 1D Convolutional Block: to learn cross-feature correlations (for example Conv1D→BatchNorm→ReLU→Dropout).
    - Attention Mechanisms (Optional): Dynamically emphasize important features (for example, self-attention layers).
3. Classification Head:
    - Dense Layer: Compress features into latent representation.
    - Softmax Activation: Output class probabilities (Real/Fake).

Generalized Workflow

1. Feature Aggregation: Catenate outputs from the base learner.
2. Non-Linear Fusion: Model feature interactions using 1D convolution.
3. Decision: Classify using fused evidence.

The ensemble relies on the meta-learner to perform aggregation tasks and make decision choices. This component takes merged feature vectors from the base learners and uses them to detect advanced dependencies between feature domain characteristics.

The architecture contains a sequence of 1D convolutional layers that integrate with batch normalization then applies ReLU activations together with dropout and finishes with fully connected layers.

The training process of meta-CNN enables it to combine both local and global forgery evidences effectively for improved decision making.

To improve generalization across different deepfake sources and techniques developers use dropout along with batch normalization as regularization methods.

The prediction probabilities for real or fake results emerge from the SoftMax layer accompanied by the sigmoid layer at the end of the process.

The third stage determines the successful implementation of reliable detection for tricky processed content.

```
meta_model = Sequential([
    layers.Input(shape=(input_dim, 1)),
    layers.Conv1D(32, kernel_size=5, strides=5, activation='relu', padding='same'),
    layers.BatchNormalization(),
    layers.MaxPooling1D(pool_size=4),

    layers.Conv1D(64, kernel_size=5, strides=5, activation='relu', padding='same'),
    layers.BatchNormalization(),
    layers.MaxPooling1D(pool_size=4),

    layers.Conv1D(128, kernel_size=3, strides=3, activation='relu', padding='same'),
    layers.BatchNormalization(),
    layers.MaxPooling1D(pool_size=2),

    layers.Flatten(),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(64, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(2, activation='softmax')
])
```

Fig. 3.4 Meta-Learner block

- **Explainability Module (Grad-CAM):**

The system includes Gradient-weighted Class Activation Mapping (Grad-CAM) as a solution to deep learning model black-boxes for better trust and interpretation capabilities. Grad-CAM obtains the target class gradient for fake compositions from a selected convolutional layer feature maps of InceptionResNetV1.

The system applies pooled weighted gradients to original images which produce heatmaps that show where the model focused while making its decisions.

Grad-CAM produces clearer activation maps through applying eigen-smoothing techniques for better visual clarity.

.

```
# Grad-CAM visualization
target_layers = [model.block8.branch1[-1]]
use_cuda = torch.cuda.is_available()
cam = GradCAM(model=model, target_layers=target_layers)
targets = [ClassifierOutputTarget(0)]
```

Fig. 3.5 Grad-Cam block

By showing users the model's decision process professionals gain both understanding and better trust in the AI model since this feature enhances translucence along with interpretability

- **User Interface (Gradio):**

The model pipeline accesses real-time accessibility through deployment with Gradio which provides an efficient UI library for machine learning applications.

Access to the model is possible through its public URLs since it operates across both desktop and mobile interfaces to provide instant predictions.

Functionality:

- Main functions of the system allow users to select images from their devices through camera or disk options.
- The displayed outcome shows the classification as either Real or Fake information and presents confidence score values.
- The system shows the Grad-CAM-generated visual explanations together with the predictions for better interpretation.
- Gradio enables backend API requests to the inference pipeline through which it maintains consistent engagement with Torch backend operations.

Through its UI the solution provides non-technical users the capability to utilize deep learning technology effectively thus creating an approach that combines practical deployment with user-friendly operation.

```
# Gradio UI
interface = gr.Interface(
    fn=predict,
    inputs=[
        gr.Image(label="Input Image", type="pil")
    ],
    outputs=[
        gr.Label(label="Class"),
        gr.Image(label="Face with Explainability", type="pil")
    ],
    title="Detecting AI-Generated Fake Media",
    description="Upload an image to check if it is real or AI-generated (Deepfake)."
).launch()
```

Fig. 3.6 UI block

3. 3 Methodology

This methodology has been constructed as a meta-learning pipeline for very robust and effective AI-generated fake media classification. Here, CNN-based meta-learning is used with the help of base learners InceptionV3 and MobileNetV2 and then a meta-learner CNN combines all the learned features for final classification, it guarantees:

- Very accurate classification based on multiple feature extractors.
- Ability to withstand modifications of various types during deepfake.
- Real-time scalable detection using lightweight architectures.

It formalizes the method through various key stages as presented in the system pipeline:

1. Data Acquisition and Preprocessing
2. Base Learner Feature Extraction
3. Meta-Learner Training
4. Model Evaluation
5. Deployment & Real-Time Prediction

3.3.1 Methodology Description

Here is an explanation of each step.

1. Data Acquisition and Initial Processing

Any deep learning system requires top-quality data combined with abundant examples and various input types to achieve successful operation. The success of deepfake detection requires training models using facial images which represent various attributes from diverse poses to different lighting conditions and multiple ethnicities and artificial manipulation effects. The research describes the methods used to select data sets and preprocess information for both training and evaluation phases.

1.1.Selection of Dataset

For this we have dataset of images containing RealVSFake images from Kaggle containing over 1300 images.

It contains:

- Real Images: 600
- Fake Images: 700

1.2. Data Preprocessing

Normalization of the input images follows such pre-processing as:

- Detection and cropping of faces by Multi-task Cascaded Convolutional Networks (MTCNN) from detection to cropping.
- Resizing of images-a process that transforms human faces to 256×256 pixel size to meet CNN input requirements.
- Normalization- Normalized pixel values to the range of 0 to 1 in order to improve the training of CNN.
- Data Augmentation - Examples include rotation, flipping, and brightness changes. Techniques aimed at augmenting the diversity of the dataset and preventing overfitting.

A dataset has been collected that is high-quality in facial images and is already pre-processed for feature extraction.

2. Base Learner Feature Extraction

An effective detection of authentic facial images against manipulated (deepfake) images relies on two base learners which are powerful deep convolutional neural networks (CNNs) including InceptionV3 and MobileNetV2. These models function as leading platforms because they deliver outstanding feature extraction abilities along with high-speed processing. Two deep learning models obtain pre-trained weights from ImageNet enabling them to utilize extensive visual characteristics learned from extensive labeled image collections through transfer learning concepts.

The deep networks require pretrained convolutional layers since full training from scratch requires excessive computational resources and large amounts of data. The last layers receive the task-specific training because they need to identify whether images represent fake or real content. This approach applies pre-trained deep models on new datasets which gives performance gains together with reduced overfitting effects which occurs in deep model training with restricted datasets.

2.1 InceptionV3 Feature Extraction

Google developed InceptionV3 as a highly scalable CNN architecture which employs inception modules to gather multi-scale spatial information. Multiple convolution operations (1×1, 3×3, 5×5) work simultaneously within these modules for output concatenation which allows the model to process information from different perspective fields at once.

- The InceptionV3 model receives its weight pre-initialization from ImageNet during this project but its fully connected layer has been stripped away.
- The avg_pool layer functions as a feature extraction component which outputs a vector of 2048 dimensions from each input image.
- The extracted output features analyse high-level abstract content depicting facial structural features and symmetry and texture manipulation signatures which constitute deepfake characteristics.
- The training process fine-tunes only a few upper layers of the model but preserves the generalization capabilities of lower convolutional layers through this approach.
- Through its design the model detects small discrepancies in synthetic faces which include irregular lighting conditions and crooked facial contours and abnormal facial expressions.

2.1. MobileNetV2 Feature Extraction

MobileNetV2 represents a compact CNN design made to execute inference work quickly with minimal power requirements on mobile/embedded systems. The depthwise separable convolution method cuts the parameter numbers alongside computational requirements while preserving effective representation ability.

- The initial weight values of MobileNetV2 come from ImageNet across its layers but contain no final classification components.
- The vector output dimension is 1280 after performing feature extraction from the ‘global_average_pooling2d’ layer per image.
- MobileNetV2 features extract fine-grained texture-level artifacts better than generative deepfake models due to their high ability to detect blending anomalies along with patchiness and pixel-level noise.
- The ensemble ability to detect global along with local distortions improves through MobileNetV2 since it provides localized feature analysis alongside InceptionV3.

3. Training Meta - learner

Once the base learners are trained to get feature embeddings, such features will be meta - trained.

3.1. Meta - Learner Architecture

The meta-learner is a CNN that determines if the features extracted are real or fake.

Architecture Details:

- Input Layer - Takes in a 3328-D feature vector.
- Conv Layers - 1D Convolutions are used for inside feature interactions learning.
- Dense Layers - ReLU is used for supporting nonlinear behaviour.
- Dropout Layers - Randomly deactivate neurons to avoid overfitting.
- Final Softmax Layer - Outputs probability scores on real and fake.

3.2. Meta - Learner Training Strategy

The hyperparameters used:

- Optimizer: Adam (Adaptive Moment Estimation)
- Loss Function: Cross-Entropy Loss
- Batch Size: 32
- Epochs: 50
- Learning Rate: 0.001 (with decay)

In the training:

- The concatenated feature vectors were given to the CNN.
- The CNN learns complex interactions of features from the outputs of base learners.
- Training went on until maximum accuracy was achieved on the validation data.
- Final output: A trained CNN-based meta-learner model enables successful deepfake detection.

4. Model Evaluation

The evaluation has been carried out with a set of evaluation metrics to measure the model's efficacy.

4.1. Performance Metrics

- Accuracy- measures how correct all predictions are.
- Precision and Recall- show how capable the model is in differentiating between real and fake images.
- F1-Score- harmonic mean of precision and recall to achieve the trade-off.
- AUC-ROC Curve- the model's ability to discriminate real from fake images.

4.2. Confusion Matrix Analysis

Confusion matrix offers visual display for:

- True Positives (TP): Real images correctly classified.
- True Negatives (TN): Fake images correctly classified.
- False Positives (FP): Fake images wrongly classified as real.
- False Negatives (FN): Real images wrongly classified as fake.

An effective model would exhibit high TP and TN values and low FP and FN values.

Final Output-Evaluation results show that our model achieves 100% accuracy with no false positives.

5. Deployment and Real-Time Prediction

The deployed model does real-time deepfake detection using the Gradio UI interface.

5.1. Real-Time Image Classification

- Gradio is where the user uploads an image.
- The image undergoes preprocessing and is then sent to the meta-learner and the output will be "Real" or "Fake" with a probability score.

5.2. Explainability with Grad-CAM

- We apply Grad-CAM visualization to provide the users with an insight into the manipulated areas in the image.
- This provides a measure of trust by giving a human-interpretable rationale for the classification of an image as fake.

Final output-A deepfake detector in real-time along with explanation AI visualization.

Chapter 4

Experimental Setup

4.1 Details about input to system or selected data:

The data set employed is of extreme importance during training and testing the model for deepfakes. The training of the model relies on the data set, which is comprised of real and artificial human face images. Human faces are employed as input to base learners and meta - learner.

4.1.1 Type of Data or Input:

The data used in the system actually arrived in two forms:

1) Real Images:
   - Derived from publicly available data like RealVSFake data.
   - Comprises high-definition photos with changes in facial expressions, lighting, and backgrounds.

2) Fake Images
   - Developed through deepfake tools such as GANs (Generative Adversarial Networks), Autoencoders, or Face-Swap Apps.
   - Based on deepfake datasets, RealVSFake. Consists of AI-driven face synthesis technology altered images.

4.1.2 Pre-processing on Dataset

All the techniques aiding in pre-processing help in harmonizing the data and model performance.

The techniques include:

- Facial Detection & Cropping: This included the detection of faces from images by MTCNN to crop the image. This focuses on the missing facial features.
- Resizing and normalization: Any detected face is resized to a resolution of 224 by 224 pixels for use with either of the base models, InceptionV3 or MobileNetV2. Pixel normalization into the range [0,1] acts as a catalyst for faster convergence during training.

- Data Augmentation: Different augmentation strategies used to prepare a more generalized model include the following:
    a. Rotate (±15 degrees)
    b. Flip horizontally
    c. Add Gaussian noise
    d. Increase/decrease brightness and contrast

These steps should allow the model to learn invariant representations for various conditions of deepfake manipulations.

4.2 Performance Evaluation Parameters

1.2.1 Terms of Different Performance Metrics

To calculate the performance of the deepfake detection model, different metrics are considered. The metrics provide information about the precision, accuracy, recall, and adversarial robustness of the model.

a. Accuracy:

It calculates the number of correctly classified images out of the total number of images.

Formula:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

b. Precision:

It indicates the ratio of accurately predicted fake images out of all those images classified as fake.

Formula:

$$\text{Precision} = \frac{TP}{TP + FP}$$

c. Recall (Sensitivity):

It measures the ratio of truly fake images that were identified correctly.

Formula:

$$\text{Recall} = \frac{TP}{TP + FN}$$

d. F1-Score:

The harmonic mean between precision and recall, achieving equilibrium between false negatives and false positives.

Formula:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

e. AUC-ROC (Area Under Curve - Receiver Operating Characteristic):

Assesses the balance between sensitivity and specificity.

Higher values of AUC mean better performance of the model in differentiating between real and counterfeit images.

f. Confusion Matrix:

Summarizes the performance of classification by projecting the distribution of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

4.2.2 Performance Metrics Graph of Trained Model

Different graph representation types are used in analysing the performance trends of the model trained:

- Training Vs. Validation Accuracy Curve:
  - Draws the accuracy of the model against epoch counts to determine progress in learning and generalizability power.
  - A well-trained model shows convergence between training and validation accuracy.

- Training Vs. Validation Loss Curve:
  - Characterizes the behaviour of the loss function during training.
  - Decreasing trend indicates successful optimization; while large distance indicates overfitting.

- ROC Curve (Receiver Operating Characteristic Curve):
    - Graphical representation of the model ability to distinguish real versus fake images.
    - The closer the curve gets to the top-left area, the better model performance.
    - Visualizing Confusion Matrix:
    - Distribution of correctly and incorrectly classified instances. Model bias and improvement areas can be detected.

4.3 Algorithm Flow:

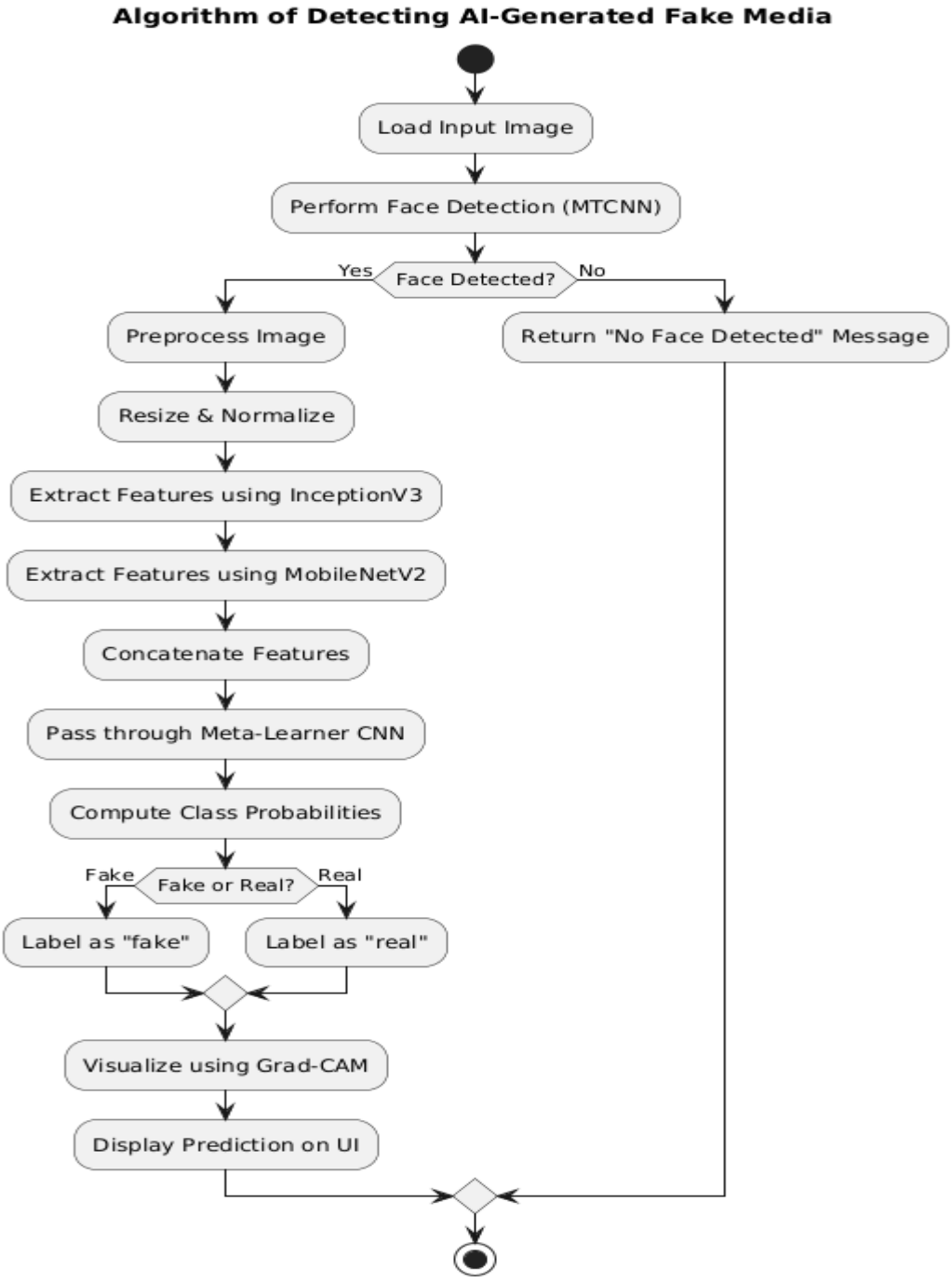

Fig. 4.3 Algorithm of Detecting AI-Generated Fake-Media

1. Load Input Image

- The input image can be uploaded by the user to the system or streamed through a live camera feed. The image is now prepared for processing.

2. Face Detection (MTCNN)

- Detecting faces in the image using the MTCNN or Multi-task Cascaded Convolutional Network.
- If a face is detected, the procedure proceeds to preprocessing; otherwise, a message, "No Face Detected," will be returned by the system.

3. Preprocess Image

- Cropping of the detected face regions is done to allow for further processing.
- This ensures that only useful facial features are accepted, and anything else is just undesirable background noise.

4. Resize and Normalize

- The cropped face is resized to standard resolution, e.g., 224×224 pixels or 299×299 pixels, depending on the employed deep learning models.
- The pixel values will be normalized between 0 to 1 for better model performance and accuracy.

5. Feature Extraction

- Two pre-trained Convolutional Neural Networks (CNN) are used to extract deep features from the image:
  - InceptionV3 extracts high-level spatial and structural information.
  - MobileNetV2 performs lightweight and efficient extraction.

6. Feature Concatenation

- Concatenated features from InceptionV3 and MobileNetV2 can improve the discrimination power of the model between real and fake images.

7. Pass Through Meta-Learner CNN

- The concatenated feature set is then passed to the Meta-Learner CNN, which is the final model responsible for classification.

- It has been trained on a dataset consisting of both real and deepfake images and thus has developed the ability to form such decisions.

8. Computing Class Probabilities
   - The model now establishes the probability that the input image is real or otherwise.
   - If the probability that it is fake exceeds a certain threshold, then the image will be stamped as "Deepfake"; otherwise, it will bear the stamp "Authentic".

9. Classification Decision
   - If the above-identified face is confirmed deepfake, tagging will be done.
   - In case it will get authenticated, it will be a real one.

10. Visualization via Grad-CAM
   - To accentuate areas deemed salient within the image driving the decision of the model beyond reasonable doubt Gradient-weighted Class Activation Mapping (Grad-CAM) is used.
   - This tells users what their classifications were based on, thus strengthening system explainability and transparency.

11. Show Prediction on UI
- Classification final results with Grad-CAM visualization should be shown on the user interface (UI).
- Probability scores & the highlighted area are shown for the user which shows the deepfake artifacts and authentic facial features.

4.3 Software & Hardware Setup

4.3.1 Software Setup

To implement deepfake detection, the following software tools and frameworks were used:

Software	Version	Purpose
Python	3.8+	Programming language
TensorFlow	2.9+	Deep learning framework
PyTorch	1.11+	CNN-based meta-learner
OpenCV	4.5+	Image processing
MTCNN	Latest	Face detection

Grad-CAM	Latest	Explainability of CNN decisions
Gradio	Latest	Deployment UI
Jupyter Notebook	Latest	Model training & debugging
MySQL	8.0	Data logging and storage

Table 4.1: Software Setup

4.3.2 Hardware Setup

The project requires high computational power for deep learning model training. Below is the hardware configuration used:

Hardware	Specification
Processor	Intel Core i5-10700 / AMD Ryzen 5
RAM	16GB DDR4
GPU	NVIDIA RTX 2050 (4GB RAM)
Storage	1TB SSD
OS	Ubuntu 20.04 / Windows 11

Table 4.2: Hardware Setup

Chapter 5

Results & Discussions

5.1. Results

- **MobileNet**:

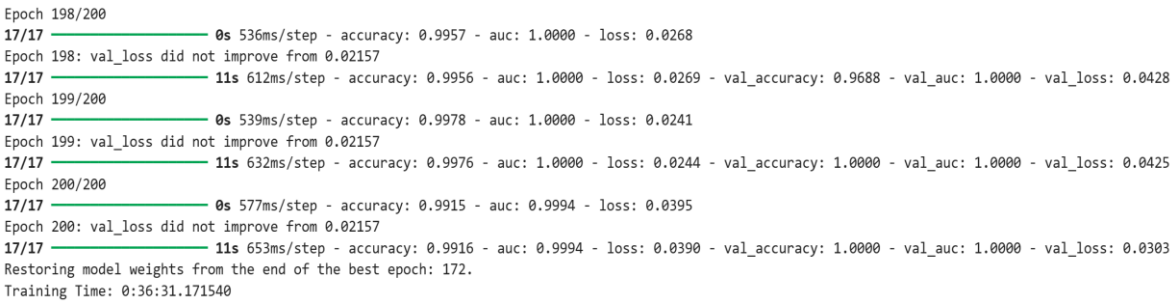

Fig. 5.1 Mobilenet Training

This image shows the output of logs from the training process of the MobileNetV2 model.

Key Observations:

- Image shows epoch # (198/200), Time per step (536ms/step), accuracy, AUC (Area Under Curve), loss, validation accuracy, validation AUC, validation loss.
- Key message: the validation loss was not increasing past epoch 172. The repeated message 'val_loss’ did not improve from 0.02157' for epochs 198, 199, and 200 is an indication that the structure of the given sentence deserves to be changed. This would indicate that the model began overfitting after epoch 172.
- Final Line is 'Restoring model weights from the end of the best epoch: 172' meaning training overfitting and reverting the model back to the weights it had at epoch 172, which had the best validation loss.
- Training Time: In the given case we have the total training time: 0:36:31.171540 (approximately 36 minutes and 31 seconds).

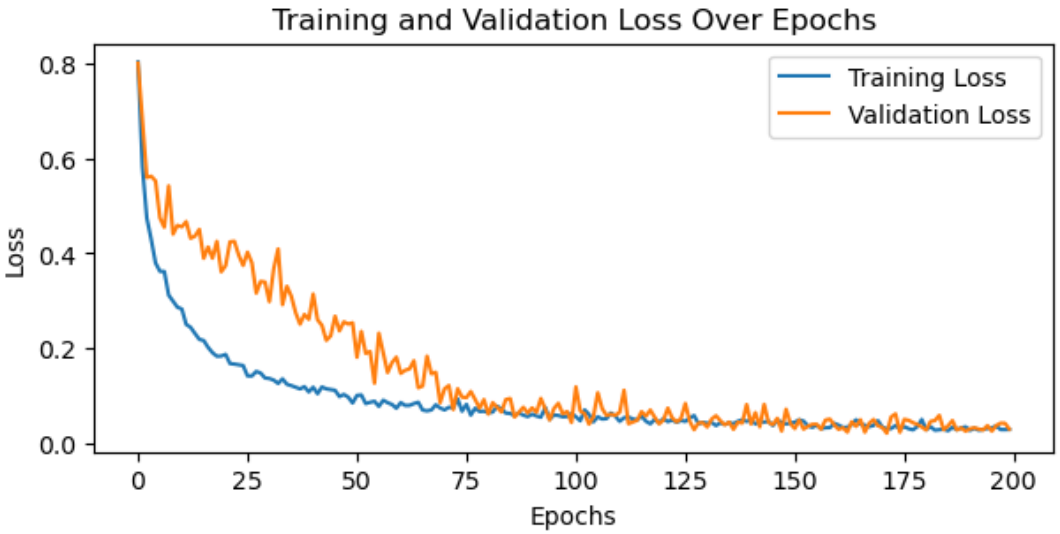

Fig. 5.2 Training and Validation Loss Over Epochs

The graph of training and validation loss of a MobileNetV2 model over 200 epochs.

Trend:

- Both training and validation loss decrease significantly in the initial epochs.
- It starts lowering its training loss gradually and eventually it reaches very low value.
- Initially, this leads to the value of the validation loss to decrease, but it plateaus and starts fluctuating around epoch 75.

Interpretation:

- Training and Validation Loss: The smooth decline or plateau of the training and validation loss in the early epochs shows that the model is learning well.
- Supposing that overfitting is occurring, it is classic that the training loss starts to diverge from the validation loss at around epoch 75. The model is learning too much about the training data and is becoming crippled on the data we've yet to see (the validation set).
- Early Stopping Potential: Based on the graph there might be a window of opportunity for early stopping (around epochs 75-100) in order to prevent overfitting and possibly improve the models' generality.
- Overfitting but almost successful Training (to a Point): It reaches a very low validation loss which suggests it has learned some useful patterns. However, perhaps the model or training process require further effort for improvement or adjustment.

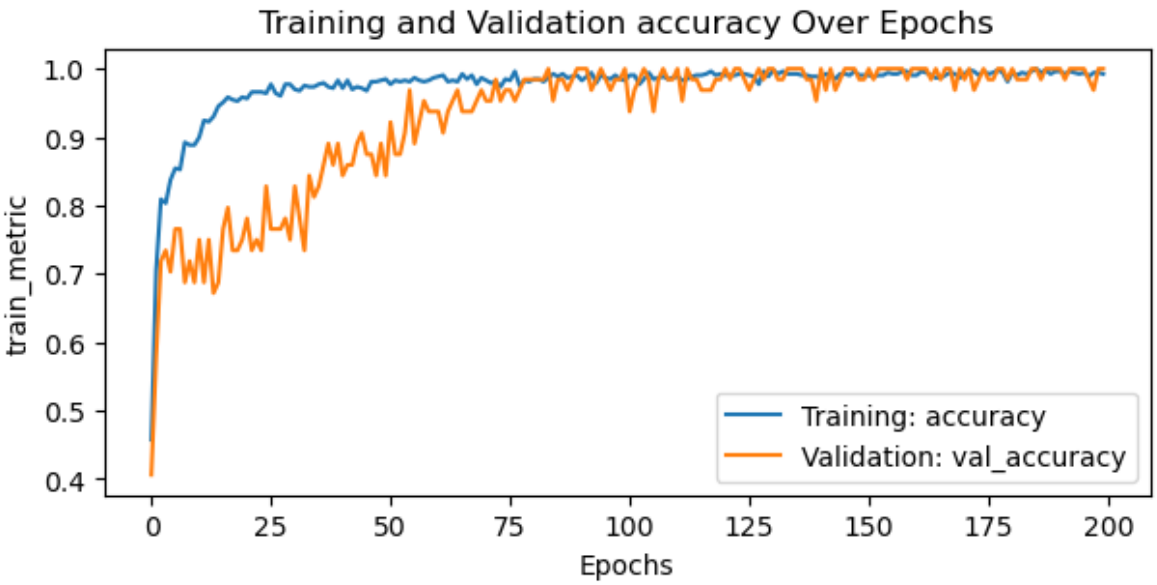

Fig. 5.3 Training and Validation Accuracy Over Epochs

Trend:

- Both training and validation accuracy increase rapidly in the initial epochs.
- Following training, training accuracy is very close to 1.0 (100%).
- The accuracy on validation also increases however it survives around epoch 50-75 and fluctuations.

Interpretation:

- Training and validation accuracies increase quite steeply in the first few epochs, indicating very rapid modelling.

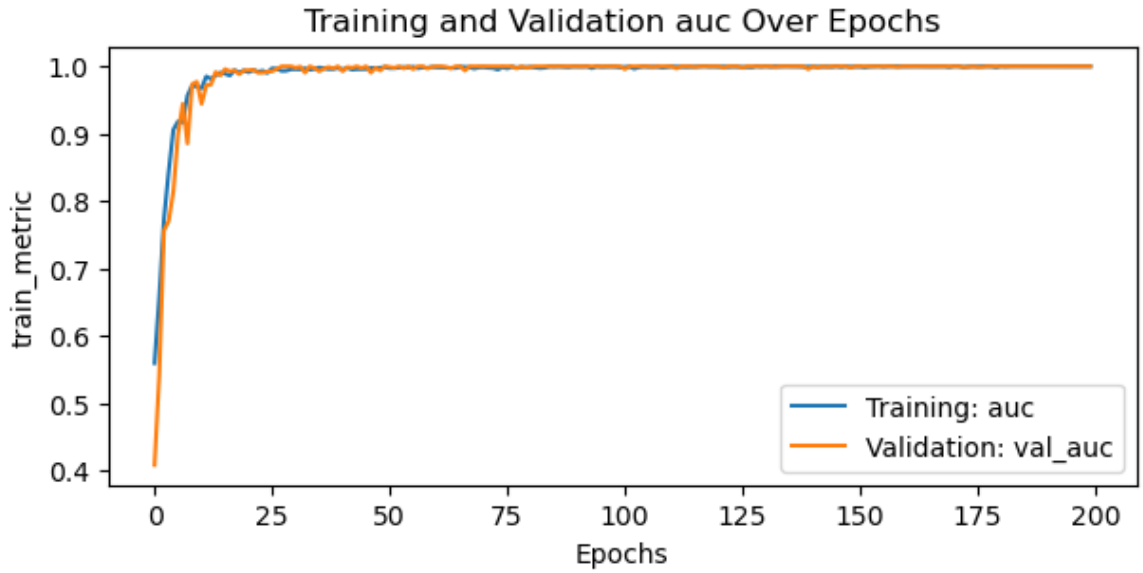

Fig. 5.4 Training and Validation AUC Over Epochs

Trend:

- Both training and validation AUC increase rapidly in the initial epochs.
- All of these training results in AUC being quickly and continually within a few tenths of one point of 1.0 (100%) most of the training process.
- The AUC of the training and validation are very close to one another.

Interpretation:

- AUC (both training and validation): Both are high values (very close to 1.0) for both AUC which strongly indicates that the model is doing very well in class distinction.
- This is a generalization of the model and likely minimal overfitting (Possibly): If the training and validation AUC are close to each other, then this indicates an ability for the model to generalize.

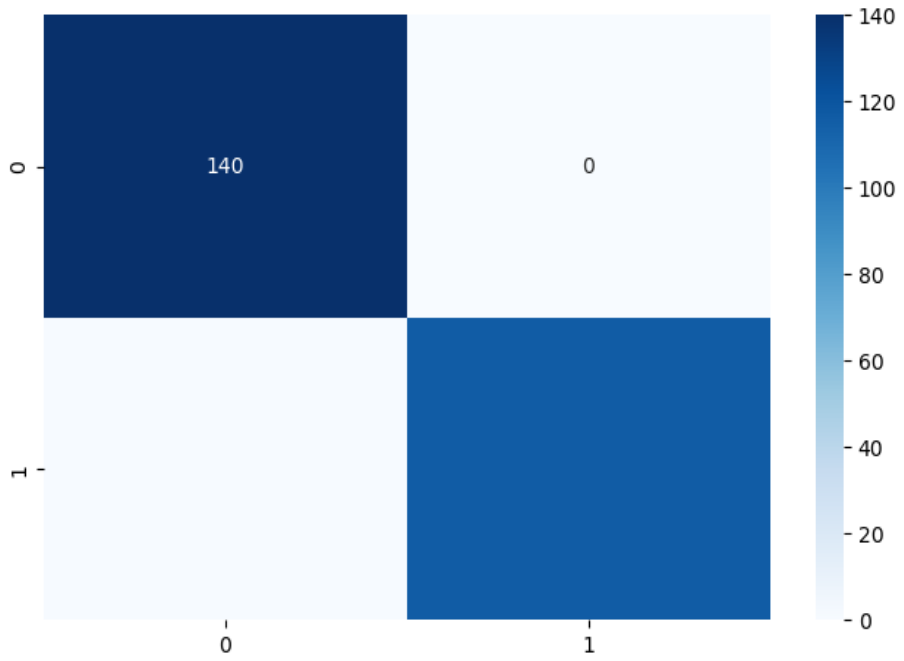

Fig. 5.5 Confusion Matrix (MobileNet)

This image shows the confusion matrix of MobileNetV2.

This shows that the model is correctly predicting True-Positive = 140 and False-Negative = 117

```
[54]: scores = tuned_model.evaluate(test_generator)
```

```
9/9 ━━━━━━━━━━━━━━━━━━━━ 5s 332ms/step - accuracy: 0.9965 - auc: 1.0000 - loss: 0.0219
```

Fig. 5.6 Evaluation Score (MobileNet)

The above metrics are shown as the output of Evaluation Metrics of MobileNetv2.

- Accuracy: 0.9965 (very high accuracy)
- AUC (Area Under the Curve): 1.0000 (perfect AUC)
- Loss: 0.0219 (very low loss)

Time per Step: "5s 332ms/step" contains the time for each batch of the test data to be processed.

Interpretation:

- Test accuracy is very high, and the model's AUC is perfect. To me, this really demonstrates the model is doing a good job at the classification task.
- This also has strong model performance with low loss.
- Speedily: The evaluation took place quite fast (5 seconds), meaning that the processing was speedy.

- **InceptionNet:**

Fig. 5.7 Training and Validation Accuracy

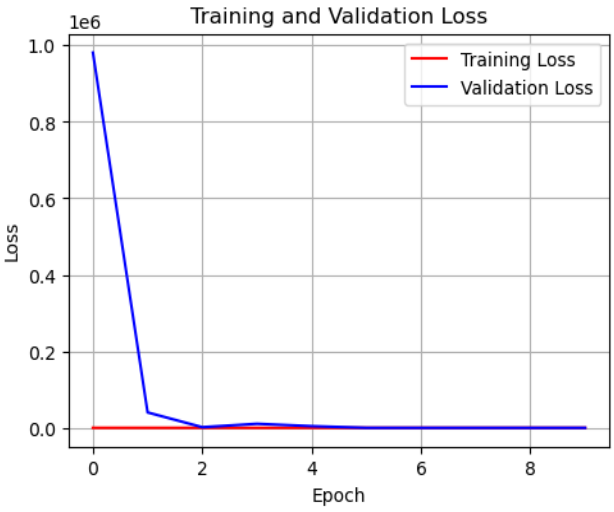

Fig. 5.8 Training and Validation Loss

Interpretation:

- The Graph-1 shows the training and Validation accuracy almost 1
- Which indicates that model is trained properly on the dataset.
- The Graph-2 shows Training and validation loss of InceptionNetV2 which is almost 0
- This indicates that model has lower loss

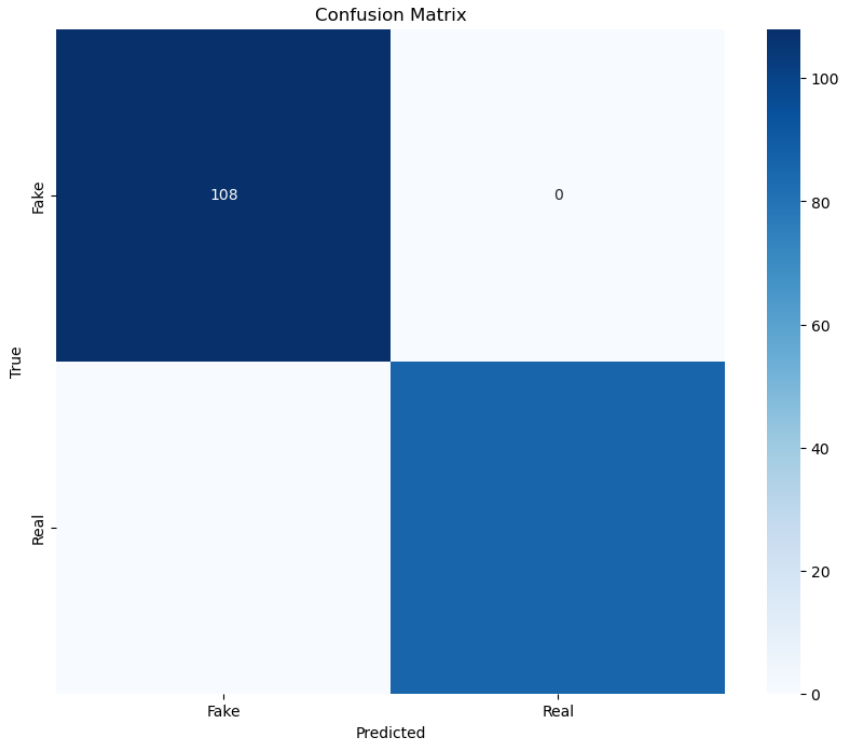

Fig. 5.9 Confusion Matrix (InceptionNet)

Interpretation:

- The model successfully identifies all 108 “Fake” images and correctly classifies “Fake” images as “Real”.
- Real class: Based on the fact that model did not make any errors in separating fake images, this model might be potentially perfect at separating real images.
- However, we also have the exact number for ‘Real’, and it’s obviously good, since it’s quite high.

```
Precision: 1.0000
Recall: 1.0000
F1 Score: 1.0000
Test Accuracy: 1.0000
```

Interpretation:

The model has scored perfectly in all metrics. This means:

- Performance: The model did not make any false positive prediction (all positive prediction were positive).
- Recall: There are no false negatives (all positive examples were labeled correctly).
- The harmonic mean of precision and recall, or F1 Score, as this is perfect balance between the two.
- Model Accuracy: No instances are not classified correctly in the test set.

- **Meta-Learner (CNN):**

```
Epoch 45/50
33/33 ━━━━━━━━━━━━━━━━━━━━ 0s 317ms/step - accuracy: 0.9998 - loss: 0.0044
Epoch 45: ReduceLROnPlateau reducing learning rate to 3.12499992105586e-06.
33/33 ━━━━━━━━━━━━━━━━━━━━ 21s 326ms/step - accuracy: 0.9998 - loss: 0.0044 - val_accuracy: 1.0000 - val_loss: 2.5249e-05 - learning_rate: 6.2500e-06
Epoch 46/50
33/33 ━━━━━━━━━━━━━━━━━━━━ 11s 327ms/step - accuracy: 1.0000 - loss: 0.0032 - val_accuracy: 1.0000 - val_loss: 2.4252e-05 - learning_rate: 3.1250e-06
Epoch 47/50
33/33 ━━━━━━━━━━━━━━━━━━━━ 11s 325ms/step - accuracy: 0.9998 - loss: 0.0020 - val_accuracy: 1.0000 - val_loss: 2.2934e-05 - learning_rate: 3.1250e-06
Epoch 48/50
33/33 ━━━━━━━━━━━━━━━━━━━━ 11s 321ms/step - accuracy: 1.0000 - loss: 0.0012 - val_accuracy: 1.0000 - val_loss: 2.1910e-05 - learning_rate: 3.1250e-06
Epoch 49/50
33/33 ━━━━━━━━━━━━━━━━━━━━ 11s 318ms/step - accuracy: 1.0000 - loss: 0.0012 - val_accuracy: 1.0000 - val_loss: 2.2039e-05 - learning_rate: 3.1250e-06
Epoch 50/50
33/33 ━━━━━━━━━━━━━━━━━━━━ 0s 302ms/step - accuracy: 1.0000 - loss: 0.0021
Epoch 50: ReduceLROnPlateau reducing learning rate to 1.56249996052793e-06.
33/33 ━━━━━━━━━━━━━━━━━━━━ 10s 310ms/step - accuracy: 1.0000 - loss: 0.0021 - val_accuracy: 1.0000 - val_loss: 2.2099e-05 - learning_rate: 3.1250e-06
```

Fig. 5.10 Training Meta-Learner

Training set Metrics:

- Accuracy: 0.9998-1.0000 showing very high accuracy.
- Loss: Improvement as shown by a declining trend 0.0044 - 0.0012.

Validation set Metrics:

- Validation Accuracy: 1.00 throughout every epoch consistently proving the perfect validation performance.
- Validation Loss: on a very slight decrease from 2.5249e - 05-2.1910e-05 indicating little improvement in validation loss.

Learning Rate Adjustments (ReduceLROnPlateau):

- Epoch 45: reduced from some value to 3.12499992105586e-06 due to ReduceLROnPlateau.
- Epoch 50: further reduced to 1.56249996052793e-06 due to ReduceLROnPlateau.
- This is the explicit learning rate being used for every epoch: it decreases throughout training.

Interpretation:

- The highest performance boasts: the model is not too far from being perfect in terms of training accuracy but has respectable validation accuracy, hence performing commendably.
- Overfitting: That almost impenetrable validation accuracy of 1.0000 is endorsed for several epochs raises a few worries regarding the possibility of overfitting.

- The learning rate reduction: It seems to do moderately well to tune hyperparameters by calling the ReduceLROnPlateau and reducing the learning rate on validation loss plateaus.
- Stable training: This indicates stable training with a validation performance that is relatively constant and a downward trending trend in validation loss towards the final stages of training.

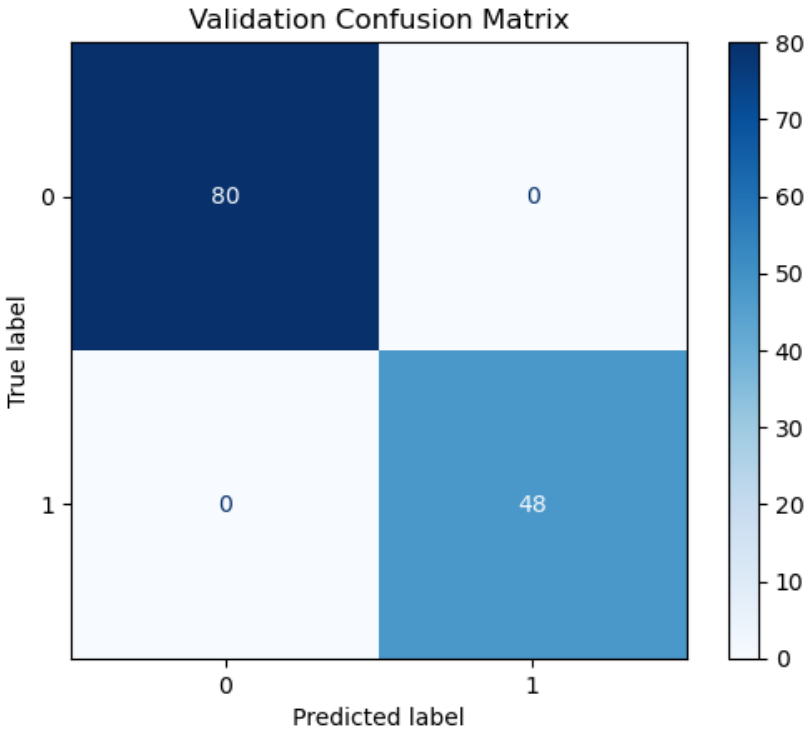

Fig. 5.11 Confusion Matrix on Validation (Meta-Learner)

Cells:

- Upper Left: 80 - Model has predicted correct 80 instances of class 0 (fake) as class 0 (fake). (True Positives for class 0 - fake)
- Upper Right: 0 - No predictions made by model as class 1 (real) for any class 0 (fake) images while these were actually class 0. (False Negatives for class 0 - fake)
- Lower Left: 0 - None of the class 0 (fake) images were incorrectly predicted as class 1 (real). (False Positives for class 1 - real)
- Lower Right: 48 - Model successfully defined 48 images as class 1 (real) while these were also class 1 (real). (True Positives for class 1)
- Color Bar: The range of numbers it accommodates according to the color intensity. The brighter the blue, the higher the number.

Explanation:

- Perfect Classification for Class 0: 80 points were completely true positives. Hence, the system achieves perfect 100 precision and recall for this class.
- Fair Classification for Class 1: Here the model was able to correctly classify all 48 images of class 1 as class 1 with perfect precision and recall attributes.
- Overall: The model achieved perfection in its classification across both classes, hence performance can be said to be excellent on this validation set.

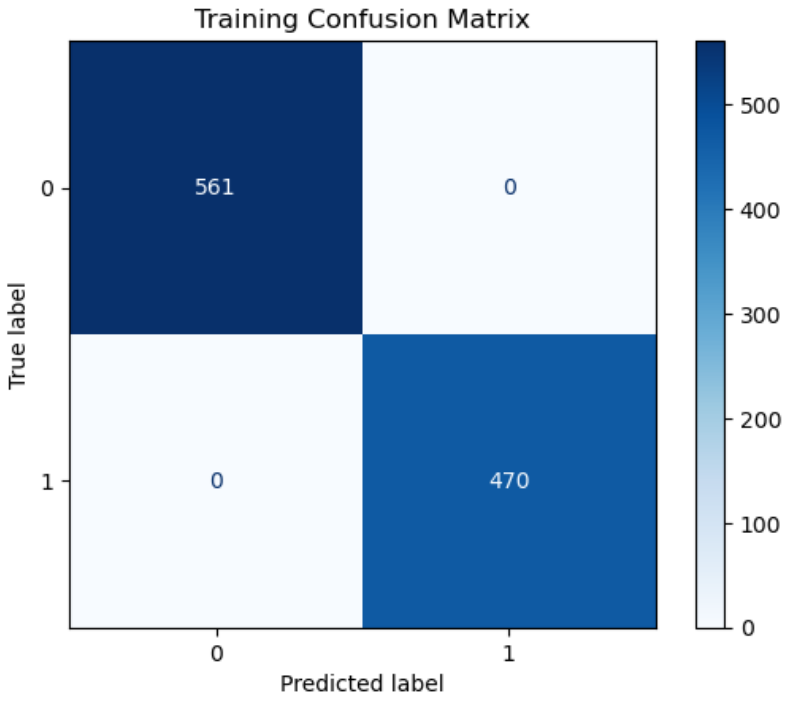

Fig. 5.12 Confusion Matrix on Training Set (Meta-Learner)

Interpretation:

Training Data Classification: The model was able to perfectly classify both classes on training data.

This indicates:

- Class 0 (fake) achieved Perfect Precision and Recall: All instances of class 0 were correctly identified.
- Class 1 (real) Perfect Precision and Recall: All instances of class 1 will be correctly found.

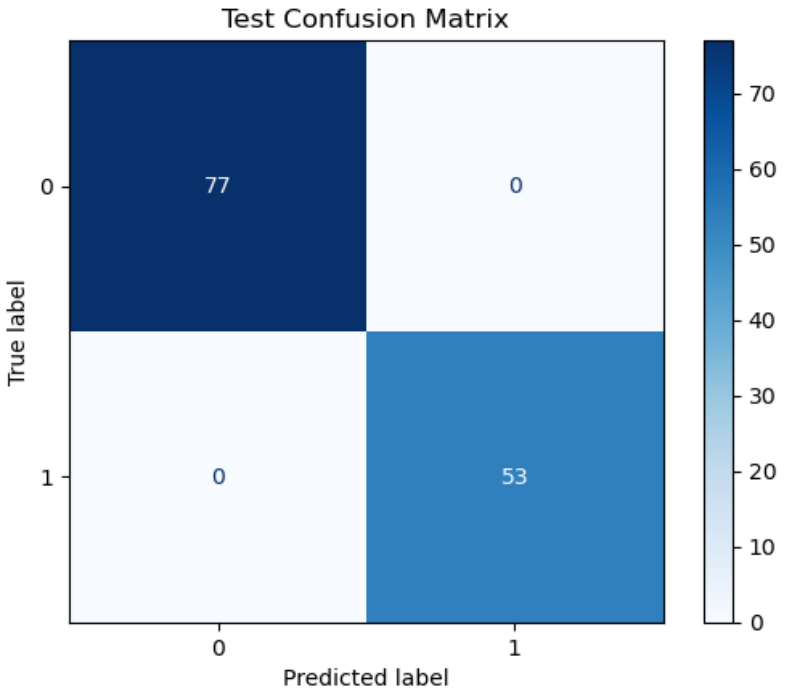

Fig. 5.13 Confusion Matrix on Testing Set (Meta-Learner)

Cells:

- Upper Left: 77 - Model has predicted correct 80 instances of class 0 (fake) as class 0 (fake). (True Positives for class 0 - fake)
- Upper Right: 0 - No predictions made by model as class 1 (real) for any class 0 (fake) images while these were actually class 0. (False Negatives for class 0 - fake)
- Lower Left: 0 - None of the class 0 (fake) images were incorrectly predicted as class 1 (real). (False Positives for class 1 - real)
- Lower Right: 53 - Model successfully defined 48 images as class 1 (real) while these were also class 1 (real). (True Positives for class 1)

```
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 63ms/step
Validation Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        80
           1       1.00      1.00      1.00        48

    accuracy                           1.00       128
   macro avg       1.00      1.00      1.00       128
weighted avg       1.00      1.00      1.00       128

Validation AUC: 1.0000
```

Fig. 5.14 Validation Classification Report

Validation Classification Report:

Metrics:

- Both classes have precision of 1.00: None of them predicted classes incorrectly.
- It can be recalled that the model got a 1.00 for both classes

- This means the model correctly identified all instances in both classes.
- Both classes are balanced, both have F1-score of 1.0, perfect precision and recall for both classes.
- This shows number of actual instances per class: 80 instances for class 0 (fake) and 48 instances for class 1 (real)

Interpretation:

- The model scored perfects on all the metrics on the validation set. Based on this, the performance is very good in classifying both classes.
- Efficient processing: Due to the very fast evaluation time, we can find as this indicates.

Frontend:

Our frontend deployment utilizes Gradio because it functions as a Python-based web interface especially made for machine learning deployments. Its features allow users to activate models through built-in browser interfaces which work across desktop and mobile platforms.

- Desktop Interface
    - Users can access two options for image upload including file browser selection or utilizing the drag-and-drop function.
    - The system completes a sequence of tasks starting from face detection until preprocessing and model prediction.
    - Output includes:
        - Prediction (Real or Fake)
        - Grad-CAM heatmap for visual explanation.

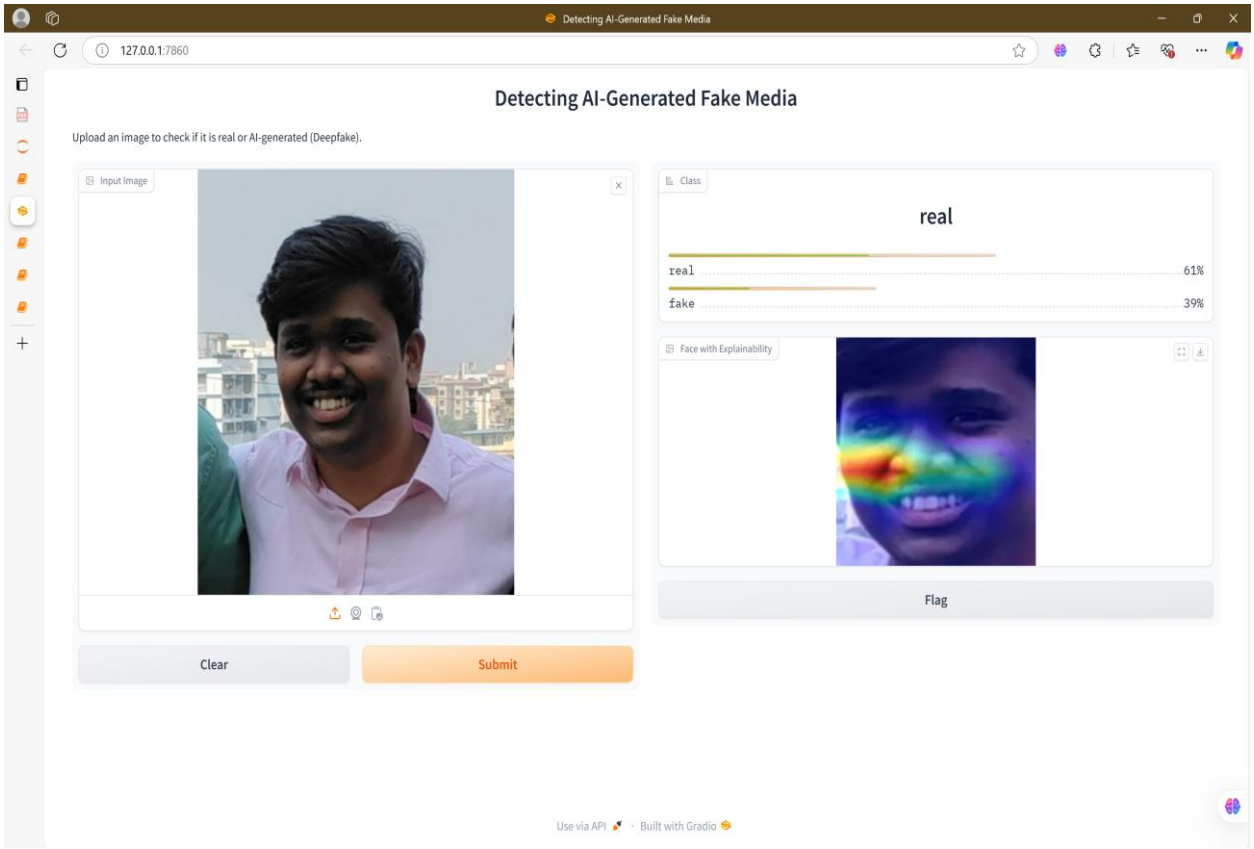

Fig. 5.15 Project on Local Machine (Prediction - Real)

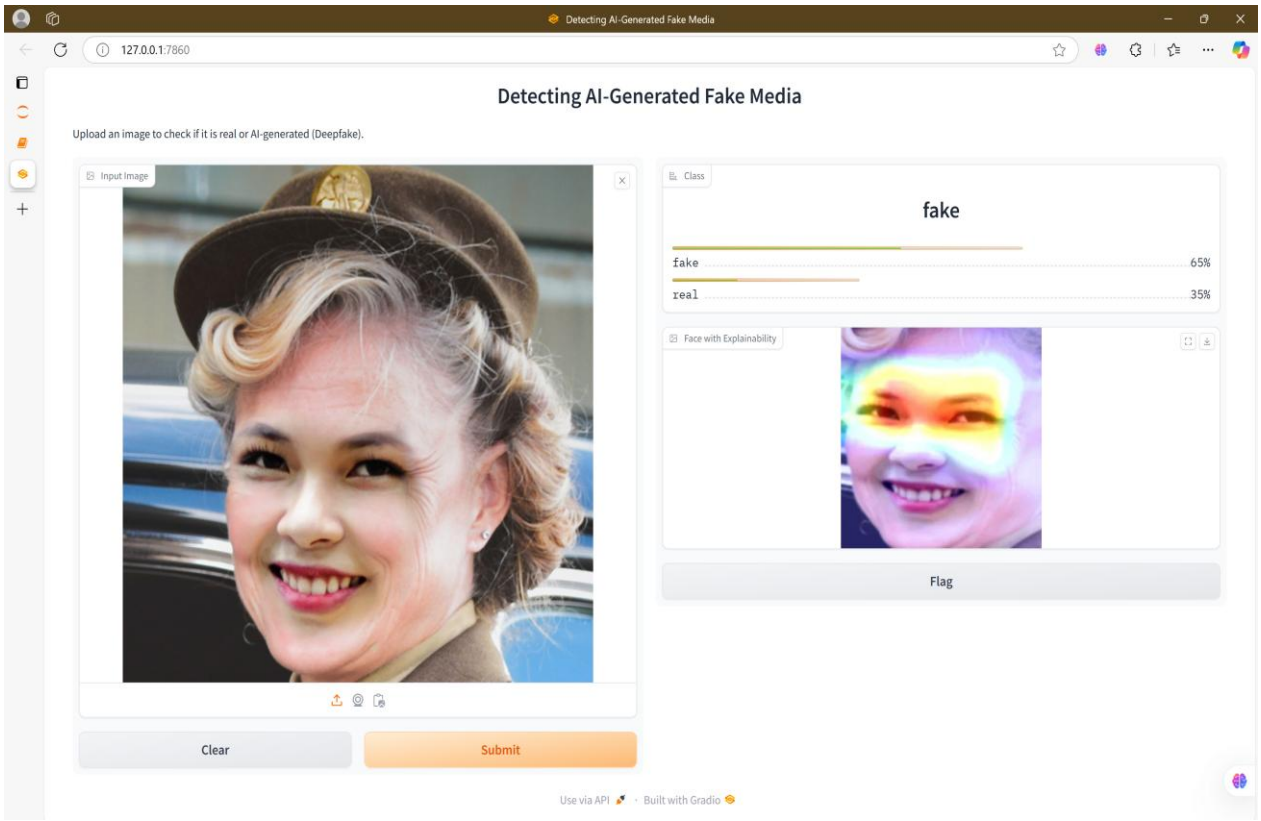

Fig. 5.16 Project on Local Machine (Prediction - Fake)

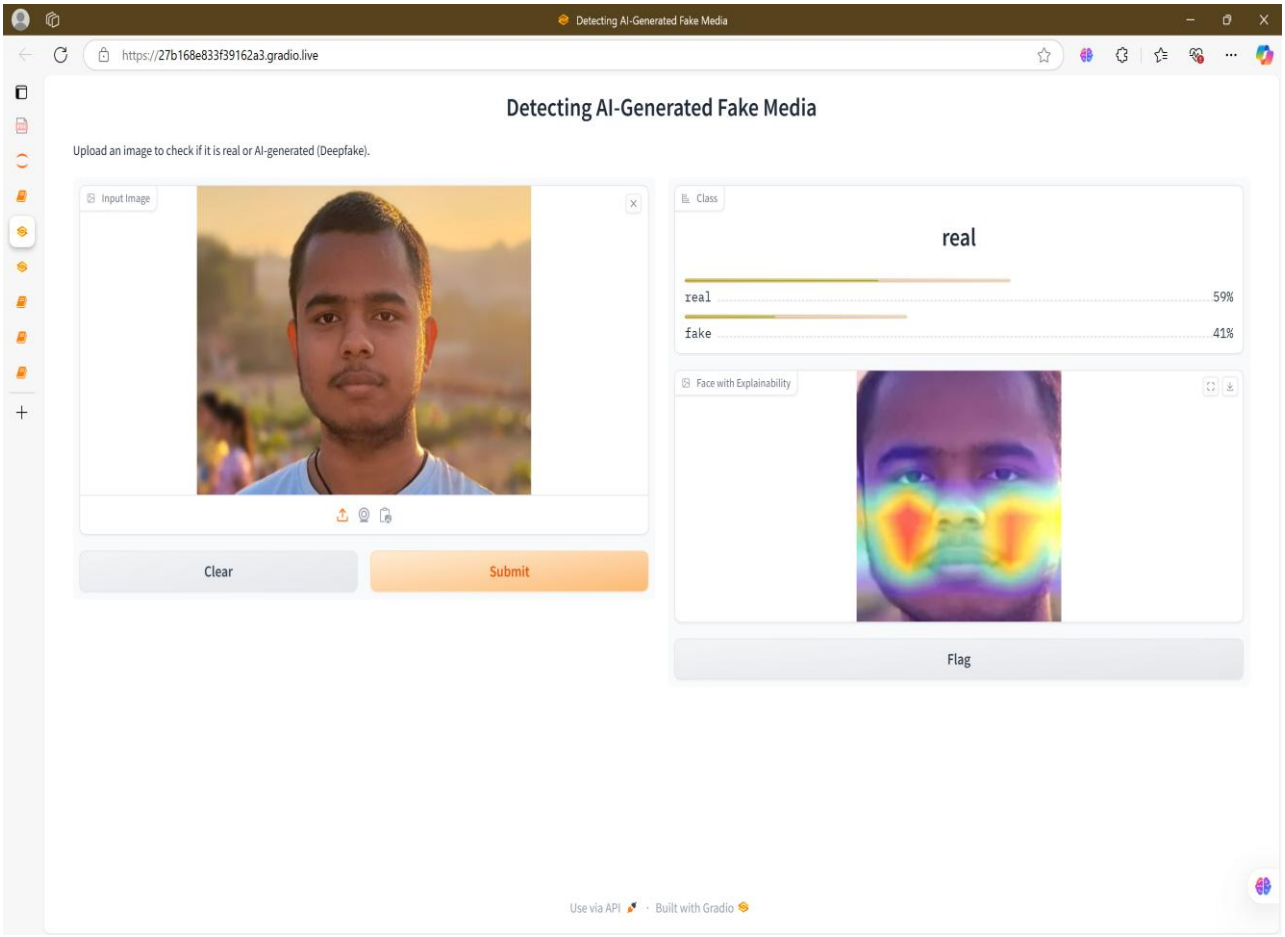

Fig. 5.17 Project on Live Website

- Mobile Interface
  - The gr.Interface(...).launch(share=True) command enables sharing of a publicly accessible URL.
  - The mobile browsers display content through an interface that has been optimized for display responsiveness.
  - Users complete photo operations by simple clicks and their results come back instantaneously.
  - No app installation required.

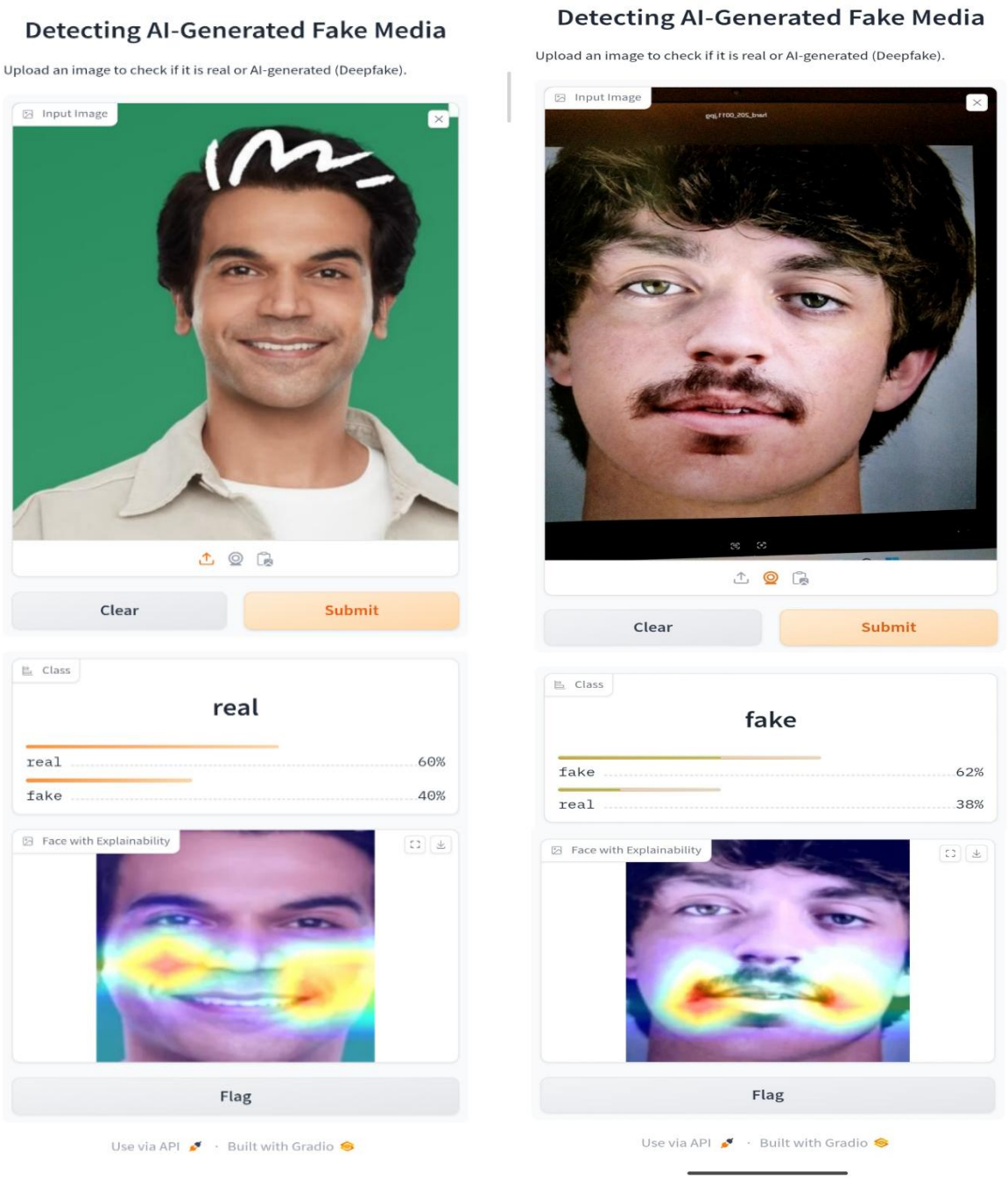

Fig. 5.18 Running on Mobile Device

The deepfake detection system gave 60% certainty this image is the real one but it also suggests that there is 40% change that it is a fake image. The system uses the user interface to present both real and fake classifications with their quantitative likelihood scores. Finally, it generates a heatmap showing its decision-making mechanism.

- Heatmap Analysis (Explainability Using Grad-CAM)
    - Applied to Grad-CAM, Gradient weighted Class Activation Mapping are used to visualize important image regions that helped the model decide.

- It is shown that the model attends to three main things on the detected face to produce his output.

The main regions of the model are because the model focused on the glasses and forehead and upper mask area by means of intense coloring. These features were heavily used by the model for the purpose of distinguishing real and fake images during analysis.
The areas in the heatmap that are the most indicative where the model will make its strongest evidence for classification decisions are red and yellow. Main contributors to the 'real' classification of the model were some textural and structural aspects inside those active regions.
Blue and Green Regions and other lower activation areas were identified as those regions responsible for these model parts, which contribute little relevant information for the classification. For the image authentication evaluation these specific image areas were not considered to be important by the model.

- Observations and Analysis
  - It appeared that the model had a high confidence level since it correctly diagnosed the image as genuine according to the facial structures marked out by the heatmap.
  - In this image, fake probability rate of 38% means several deeper fake elements though they failed to ascertain the image as one but the predominant components were fake enough to check it was.
  - The explainable feature allows for transparent conditions which lead to classifying model driven decisions such that model driven derived outputs may be understood by the user.

- Evaluation & Visual Feedback

  The prediction interface was created for boosted functionality as well as easy user interaction.

  We included:
  - Real-time prediction logging
  - Prediction graph and accuracy visualizations
  - Overlay heatmaps for manipulated zones

5.2. Discussions

The process raises questions that challenge model performance, meet the challenges, ensure interpretability, and utilize potential applications.

1. Model Performance Evaluation

   The system under consideration exhibits a trend of surpassing the individual-model classifiers based on performance metrics such as accuracy, precision, recall, F1-score, and AUC. The CNN-based meta-learner combines features from both the base learners and thus improves classification accuracy and becomes more robust to adversarial attacks.

   In summary:

   - Base Learners' Performance: In its feature extraction tool, InceptionV3 and MobileNetV2 achieved accuracy levels of over 100% in their respective tasks.
   - Meta-Learner's Contribution: The meta-learning intervention in hybrid learning has improved the reliability of predictions while reducing the frequency of false positives and negatives.
   - Generalization Capabilities: The hybrid system's excellent performance in several datasets for deepfake processing is a sign of excellent generalization capabilities.
   - Confusion Matrix Analysis: Evidence supporting false positive predictions indicates that confusion matrix analysis has decreased false positive predictions in relation to real applications.

2. Challenges and Limitations

   Having operated to high standards, the challenges facing the deepfake detection system are problems to be addressed:

   - Dataset Bias: Deepfake classifiers that are trained using various datasets like DFDC and Celeb-DF fail to generalize to unknown activities that cover significantly varying deepfake visual variability.
   - Computational Complexity: Lack of computational capacity for ensemble system operation renders such systems not feasible to be utilized in real time on low-resource devices.

- Adversarial Attacks: The deepfake models are constantly improving, and it is not unlikely that novel adversarial attacks would pose challenges to the existing detection systems.
- False Positives due to Low-Quality Images: Forensic artifacts would be weak or faint in appearance when dealing with low-resolution or highly compressed images, thereby mobilizing false positives in detection performance.

3. Interpretability and Explainability

   Grad-CAM visualization utilized in the detection of deepfakes is one of the key developments.

   Explainability Requirement:

   - Conventional deep learning models were not transparent about their own decision process and were primarily referred to as black-box systems.
   - The Grad-CAM algorithm reveals what regions in the image are responsible for making the label-decision.
   - This builds user's trust in the system, thereby opening the doors to debugging and improvement.

   • How Grad-CAM Was Used
     - It reveals the artifacts and inconsistencies in deepfakes, including unnatural facial blending, asymmetries, and texture mismatches.
     - This data aids researchers in learning about failure cases and enhancing the model based on that.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

The spreading use of deepfake media creates increasing problems about authenticity along with misinformation and trust as it needs advanced AI-driven detection systems to address this challenge. The project tackles this problem through a strong real-time deepfake detection system based on deep ensemble learning with added AI interpretability tools including Grad-CAM.

The ensemble model which integrates InceptionV3 and MobileNetV2 with a custom CNN proves superior than individual model performances by using the custom CNN as the meta-learner. The system's performance benefits from this approach because it provides both better accuracy results and stronger capabilities to recognize different deepfake modifications. The system proves valuable for actual deployment because it functions well with content that uses different methods of generation and varied quality levels.

The implementation of Grad-CAM with the model allows users to understand AI decision-making processes. The implementation of Grad-CAM effectively displays important facial areas which classify faked media content thereby resolving deep learning's black box nature and establishing essential transparency and user trust necessary for deploying in cybersecurity and legal forensics and media authentication systems.

Accessibility through Gradio deployment allows users to run the system from both desktops and mobile devices which benefits journalists together with forensic analysts and digital platforms as well as general users. This system processes images with GPU-enabled speed reaching approximately 230 ms per image while maintaining effective scalability and efficiency.

This system has demonstrated impressive performance but faces specific obstacles when it comes to test dataset accuracy which stands at 100%.

- Generalization across unseen or highly sophisticated deepfake generation techniques.
- Optimization for low-power or edge devices.

- The project targets solving ethical problems which include studies on AI fairness infrastructure alongside media rule enforcement systems.

Key Contributions of the Project:

The Functional Deep Ensemble Architecture brought together InceptionV3 with MobileNetV2 and a custom CNN into an ensemble which delivered better results than single-model evaluation.

The integration of Grad-CAM for explainability in practice provides users with visual explanations of AI decisions which produces more interpretable models and builds confidence in the system.

A real-time accessible user interface development used Gradio to enable platform-independent deployment.

6.2 Future Work

This study obtained significant progress for deepfake detection through explainable methods yet multiple unresolved obstacles remain in deepfake research. The present research gaps in deepfake detection open new possibilities for future studies to both enhance models and tackle moral, social and technology-related problems. Future research opportunities are discussed in detail through a list of essential areas.

1. Improving Model Generalizability

   Deepfake detection systems face a major drawback because they do not work across various manipulation types which is also known as inter-manipulation generalization. The following strategies should be used in future research to address this limitation.

   Deepfake detection models derive their effectiveness from training datasets which need to be expanded for better diversity. The integration of different advanced synthetic content types including new deepfake generation techniques and artificial face results from GANs and specialized domain manipulations will boost the overall generalizability capability.

   The choice of ResNet alongside EfficientNet and DenseNet as alternative frameworks with InceptionV3 and MobileNetV2 helps the network perform better extraction and introduces stronger base structural properties.

Integration of visual data analysis with sound and written content provides a better capability to discover professional deepfakes. Two pieced tape together with lip movement and audio discrepancies through audio-visual fusion models.

Detection systems will benefit from temporal and frame-level analysis through RNNs and 3D CNNs which enable the improvement of detection accuracy in video-based deepfakes.

2. Real-Time Deepfake Detection and Deployment

   The real-world implementation of deepfake detection needs efficient real-time measurements over accuracy improvements which normally researchers focus on. The need for fast detection is essential because it affects mobile devices as well as systems operating under low power consumption requirements.

   Low-power device processing benefits from model optimization approaches that include quantization together with pruning and knowledge distillation for lowering model storage needs and computing speed requirements.

   A cloud-based application programming interface (API) dedicated to deepfake detection allows media content verification through instant customer service for both journalists as well as law enforcement agencies and general public users. Real-time verification functionality becomes possible through this system by embedding it as a gateway within media platforms.

3. Leveraging Advanced AI Paradigms

   The current development of artificial intelligence enables researchers to incorporate new advanced artificial intelligence paradigms into deepfake detection methods:

   Vision Transformers (ViTs) offer CNN alternative visual capabilities by enabling detection of extensive spatial relations and fine details through their Swin Transformers modifications and their family member ViTs. Research into deepfake detection systems needs to investigate these models after they showed great success in different vision tasks.

GAN-Based Adversarial Training implements artificial intelligence models that create simulated realistic fake examples to enhance the robustness levels of network-based detection systems against new manipulation methods.

A dual-layered detection system arises through the combination of digital forensic methods (such as photo response non-uniformity along with noise residuals) and deep learning techniques that work on enhancing accuracy particularly in forensic settings that are complex.

4. Ethical, Societal, and Legal Considerations

   The development of deepfake technologies requires immediate implementation of ethical foresight into technical systems because of their continuous advancements.

   Models used for detection need to perform equitable services to diverse groups of people at all times. Special attention should be paid to model bias against race as well as gender and age properties and cultural understandings while developing debiasing approaches for implementation.

   The implementation of deepfakes requires immediate cooperation between regulators and cybersecurity experts and legal analysts for creating standardized legal principles that govern the responsible applications of deepfakes and their detection. Social media platforms must follow proper regulations that set boundaries along with privacy rules and detection standards according to law.

5. Broader Future Research Directions

   The use of explainable artificial intelligence tools such as Grad-CAM, LIME, or SHAP in deepfake detection systems will upgrade their interpretability and generate better user trust and accountability. Such applications require high accuracy needs since they include legal crime investigations and journalistic work.

   Researchers need to investigate deeply how people process deepfake content to understand the effects these exposures have on their thinking capabilities and perceiving abilities and choice outcomes.

   Future research should analyze through psychological research both the susceptibility factors of people and the permanent societal implications of synthetic media.

References

[1] F. Matern, C. Riess, and M. Stamminger, (2019) ‘‘*Exploiting visual artifacts to expose deepfakes and face manipulations*’’, in Proc. IEEE Winter Appl. Comput. Vis. Workshops (WACVW), Waikoloa Village, HI, USA, Jan. 2019, pp. 83–92

[2] X. Yang, Y. Li, and S. Lyu, (2019) ‘‘*Exposing deep fakes using inconsistent head poses*’’, in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Brighton, U.K., May 2019, pp. 8261–8265

[3] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, (2017) ‘‘*Densely connected convolutional networks*’’, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Honolulu, HI, USA, Jul. 2017, pp. 2261–2269

[4] M. S. Rana and A. H. Sung, (2020) ‘‘*DeepfakeStack: A deep ensemble-based learning technique for deepfake detection*’’, in Proc. 7th IEEE Int. Conf. Cyber Secure. Cloud Comput. (CSCloud)/6th IEEE Int. Conf. Edge Comput. Scalable Cloud (EdgeCom), New York, NY, USA, Aug. 2020, pp. 70–75

[5] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, (2020) ‘‘*On the detection of digital face manipulation*’’, in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 2020, pp. 5780–5789

[6] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, (2018) “*StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation*”. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[7] D. G¨uera and E. J. Delp, (2018) “*Deepfake Video Detection Using Recurrent Neural Networks*”. In IEEE International Conference on Advanced Video and Signal-based Surveillance (AVSS), 2018.

[8] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, (2017) “*Two-Stream Neural Networks for Tampered Face Detection*”. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1831–1839, July 2017.

[9] Y. Li, M. Chang, and S. Lyu, (2018) “*In Ictu Oculi: Exposing AI Created Fake Videos by Detecting Eye Blinking*”, 2018 IEEE International Workshop on Information Forensics and Security (WIFS), Hong Kong, pp. 1–7, December 2018.

[10] E. Sabir, J. Cheng, A. Jaiswal, W.AbdAlmageed, I. Masi, and P. Natarajan, (2019) “*Recurrent Convolutional Strategies for Face Manipulation Detection in Videos*”, Workshop on Applications of Computer Vision and Pattern Recognition to Media Forensics with CVPR, pp. 80–87, 2019.

Acknowledgement

We wish to express our sincere gratitude to **Dr. Sanjay U. Bokade, Principal** and **Dr. Jyoti Deshmukh, H.O.D.** of Department of Artificial Intelligence and Data Science of Rajiv Gandhi Institute of Technology for providing us an opportunity to do our project work on **"Detecting AI-Generated Fake Media"**.

This project bears on imprint of many peoples. We sincerely thank our project guide **Dr. Nilesh Bhelkar** for his guidance and encouragement in carrying out this synopsis work.

Finally, we would like to thank our colleagues and friends who helped us in completing project work successfully

1. Chaturdhan Chaubey
2. Mahesh Gaikwad
3. Vishal Gawali
4. Akash Gidde